



**The 52nd Annual Meeting of the
Association for Computational Linguistics**

**Proceedings of the Conference
Volume 2: Short Papers**

**ACL 2014
June 22–27
Baltimore**

Platinum Level Sponsor:



Gold Level Sponsors:



Silver Level Sponsors:



Bronze Level Sponsors:



Supporters:



©2014 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

ISBN 978-1-937284-73-2

Table of Contents

<i>Exploring the Relative Role of Bottom-up and Top-down Information in Phoneme Learning</i> Abdellah Fourtassi, Thomas Schatz, Balakrishnan Varadarajan and Emmanuel Dupoux	1
<i>Biases in Predicting the Human Language Model</i> Alex B. Fine, Austin F. Frank, T. Florian Jaeger and Benjamin Van Durme	7
<i>Probabilistic Labeling for Efficient Referential Grounding based on Collaborative Discourse</i> Changsong Liu, Lanbo She, Rui Fang and Joyce Y. Chai	13
<i>A Composite Kernel Approach for Dialog Topic Tracking with Structured Domain Knowledge from Wikipedia</i> Seokhwan Kim, Rafael E. Banchs and Haizhou Li	19
<i>An Extension of BLANC to System Mentions</i> Xiaoqiang Luo, Sameer Pradhan, Marta Recasens and Eduard Hovy	24
<i>Scoring Coreference Partitions of Predicted Mentions: A Reference Implementation</i> Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng and Michael Strube 30	
<i>Measuring Sentiment Annotation Complexity of Text</i> Aditya Joshi, Abhijit Mishra, Nivvedan Senthamilselvan and Pushpak Bhattacharyya	36
<i>Improving Citation Polarity Classification with Product Reviews</i> Charles Jochim and Hinrich Schütze	42
<i>Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification</i> Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou and Ke Xu	49
<i>Sprinkling Topics for Weakly Supervised Text Classification</i> Swapnil Hingmire and Sutanu Chakraborti	55
<i>A Feature-Enriched Tree Kernel for Relation Extraction</i> Le Sun and Xianpei Han	61
<i>Employing Word Representations and Regularization for Domain Adaptation of Relation Extraction</i> Thien Huu Nguyen and Ralph Grishman	68
<i>Graph Ranking for Collective Named Entity Disambiguation</i> Ayman Alhelbawy and Robert Gaizauskas	75
<i>Descending-Path Convolution Kernel for Syntactic Structures</i> Chen Lin, Timothy Miller, Alvin Kho, Steven Bethard, Dmitriy Dligach, Sameer Pradhan and Guergana Savova	81
<i>Entities' Sentiment Relevance</i> Zvi Ben-Ami, Ronen Feldman and Binyamin Rosenfeld	87
<i>Automatic Detection of Multilingual Dictionaries on the Web</i> Gintare Grigonyte and Timothy Baldwin	93
<i>Automatic Detection of Cognates Using Orthographic Alignment</i> Alina Maria Ciobanu and Liviu P. Dinu	99

<i>Automatically constructing Wordnet Synsets</i>	
Khang Nhut Lam, Feras Al Tarouti and Jugal Kalita	106
<i>Constructing a Turkish-English Parallel TreeBank</i>	
Olcay Taner Yıldız, Ercan Solak, Onur Görgün and Raziieh Ehsani	112
<i>Improved Typesetting Models for Historical OCR</i>	
Taylor Berg-Kirkpatrick and Dan Klein	118
<i>Robust Logistic Regression using Shift Parameters</i>	
Julie Tibshirani and Christopher D. Manning	124
<i>Faster Phrase-Based Decoding by Refining Feature State</i>	
Kenneth Heafield, Michael Kayser and Christopher D. Manning	130
<i>Decoder Integration and Expected BLEU Training for Recurrent Neural Network Language Models</i>	
Michael Auli and Jianfeng Gao	136
<i>On the Elements of an Accurate Tree-to-String Machine Translation System</i>	
Graham Neubig and Kevin Duh	143
<i>Simple extensions and POS Tags for a reparameterised IBM Model 2</i>	
Douwe Gelling and Trevor Cohn	150
<i>Dependency-based Pre-ordering for Chinese-English Machine Translation</i>	
Jingsheng Cai, Masao Utiyama, Eiichiro Sumita and Yujie Zhang	155
<i>Generalized Character-Level Spelling Error Correction</i>	
Noura Farra, Nadi Tomeh, Alla Rozovskaya and Nizar Habash	161
<i>Improved Iterative Correction for Distant Spelling Errors</i>	
Sergey Gubanov, Irina Galinskaya and Alexey Baytin	168
<i>Predicting Grammaticality on an Ordinal Scale</i>	
Michael Heilman, Aoife Cahill, Nitin Madnani, Melissa Lopez, Matthew Mulholland and Joel Tetreault	174
<i>I'm a Belieber: Social Roles via Self-identification and Conceptual Attributes</i>	
Charley Beller, Rebecca Knowles, Craig Harman, Shane Bergsma, Margaret Mitchell and Benjamin Van Durme	181
<i>Automatically Detecting Corresponding Edit-Turn-Pairs in Wikipedia</i>	
Johannes Daxenberger and Iryna Gurevych	187
<i>Two Knives Cut Better Than One: Chinese Word Segmentation with Dual Decomposition</i>	
Mengqiu Wang, Rob Voigt and Christopher D. Manning	193
<i>Effective Document-Level Features for Chinese Patent Word Segmentation</i>	
Si Li and Nianwen Xue	199
<i>Word Segmentation of Informal Arabic with Domain Adaptation</i>	
Will Monroe, Spence Green and Christopher D. Manning	206
<i>Resolving Lexical Ambiguity in Tensor Regression Models of Meaning</i>	
Dimitri Kartsaklis, Nal Kalchbrenner and Mehrnoosh Sadrzadeh	212

<i>A Novel Content Enriching Model for Microblog Using News Corpus</i> Yunlun Yang, Zhihong Deng and Hongliang Yu	218
<i>Learning Bilingual Word Representations by Marginalizing Alignments</i> Tomáš Kočiský, Karl Moritz Hermann and Phil Blunsom	224
<i>Detecting Retries of Voice Search Queries</i> Rivka Levitan and David Elson	230
<i>Sliding Alignment Windows for Real-Time Crowd Captioning</i> Mohammad Kazemi, Rahman Lavaee, Iftekhar Naim and Daniel Gildea	236
<i>Detection of Topic and its Extrinsic Evaluation Through Multi-Document Summarization</i> Yoshimi Suzuki and Fumiyo Fukumoto	241
<i>Content Importance Models for Scoring Writing From Sources</i> Beata Beigman Klebanov, Nitin Madnani, Jill Burstein and Swapna Somasundaran	247
<i>Chinese Morphological Analysis with Character-level POS Tagging</i> Mo Shen, Hongxiao Liu, Daisuke Kawahara and Sadao Kurohashi	253
<i>Part-of-Speech Tagging using Conditional Random Fields: Exploiting Sub-Label Dependencies for Improved Accuracy</i> Miikka Silfverberg, Teemu Ruokolainen, Krister Lindén and Mikko Kurimo	259
<i>POS induction with distributional and morphological information using a distance-dependent Chinese restaurant process</i> Kairit Sirts, Jacob Eisenstein, Micha Elsner and Sharon Goldwater	265
<i>Improving the Recognizability of Syntactic Relations Using Contextualized Examples</i> Aditi Muralidharan and Marti A. Hearst	272
<i>How to Speak a Language without Knowing It</i> Xing Shi, Kevin Knight and Heng Ji	278
<i>Assessing the Discourse Factors that Influence the Quality of Machine Translation</i> Junyi Jessy Li, Marine Carpuat and Ani Nenkova	283
<i>Automatic Detection of Machine Translated Text and Translation Quality Estimation</i> Roe Aharoni, Moshe Koppel and Yoav Goldberg	289
<i>Improving sparse word similarity models with asymmetric measures</i> Jean Mark Gawron	296
<i>Dependency-Based Word Embeddings</i> Omer Levy and Yoav Goldberg	302
<i>Vector spaces for historical linguistics: Using distributional semantics to study syntactic productivity in diachrony</i> Florent Perex	309
<i>Single Document Summarization based on Nested Tree Structure</i> Yuta Kikuchi, Tsutomu Hirao, Hiroya Takamura, Manabu Okumura and Masaaki Nagata	315
<i>Linguistic Considerations in Automatic Question Generation</i> Karen Mazidi and Rodney D. Nielsen	321

<i>Polynomial Time Joint Structural Inference for Sentence Compression</i>	
Xian Qian and Yang Liu	327
<i>A Bayesian Method to Incorporate Background Knowledge during Automatic Text Summarization</i>	
Annie Louis	333
<i>Predicting Power Relations between Participants in Written Dialog from a Single Thread</i>	
Vinodkumar Prabhakaran and Owen Rambow	339
<i>Tri-Training for Authorship Attribution with Limited Training Data</i>	
Tieyun Qian, Bing Liu, Li Chen and Zhiyong Peng	345
<i>Automation and Evaluation of the Keyword Method for Second Language Learning</i>	
Gözde Özbal, Daniele Pighin and Carlo Strapparava	352
<i>Citation Resolution: A method for evaluating context-based citation recommendation systems</i>	
Daniel Duma and Ewan Klein	358
<i>Hippocratic Abbreviation Expansion</i>	
Brian Roark and Richard Sproat	364
<i>Unsupervised Feature Learning for Visual Sign Language Identification</i>	
Binyam Gebrekidan Gebre, Onno Crasborn, Peter Wittenburg, Sebastian Drude and Tom Heskes	370
<i>Experiments with crowdsourced re-annotation of a POS tagging data set</i>	
Dirk Hovy, Barbara Plank and Anders Søgaard	377
<i>Building Sentiment Lexicons for All Major Languages</i>	
Yanqing Chen and Steven Skiena	383
<i>Difficult Cases: From Data to Learning, and Back</i>	
Beata Beigman Klebanov and Eyal Beigman	390
<i>The VerbCorner Project: Findings from Phase 1 of crowd-sourcing a semantic decomposition of verbs</i>	
Joshua K. Hartshorne, Claire Bonial and Martha Palmer	397
<i>A Corpus of Sentence-level Revisions in Academic Writing: A Step towards Understanding Statement Strength in Communication</i>	
Chenhao Tan and Lillian Lee	403
<i>Determiner-Established Deixis to Communicative Artifacts in Pedagogical Text</i>	
Shomir Wilson and Jon Oberlander	409
<i>Modeling Factuality Judgments in Social Media Text</i>	
Sandeep Soni, Tanushree Mitra, Eric Gilbert and Jacob Eisenstein	415
<i>A Topic Model for Building Fine-grained Domain-specific Emotion Lexicon</i>	
Min Yang, Dingju Zhu and Kam-Pui Chow	421
<i>Depeche Mood: a Lexicon for Emotion Analysis from Crowd Annotated News</i>	
Jacopo Staiano and Marco Guerini	427
<i>Improving Twitter Sentiment Analysis with Topic-Based Mixture Modeling and Semi-Supervised Training</i>	
Bing Xiang and Liang Zhou	434

<i>Cross-cultural Deception Detection</i>	
Verónica Pérez-Rosas and Rada Mihalcea	440
<i>Particle Filter Rejuvenation and Latent Dirichlet Allocation</i>	
Chandler May, Alex Clemmer and Benjamin Van Durme	446
<i>Comparing Automatic Evaluation Measures for Image Description</i>	
Desmond Elliott and Frank Keller	452
<i>Learning a Lexical Simplifier Using Wikipedia</i>	
Colby Horn, Cathryn Manduca and David Kauchak	458
<i>Cheap and easy entity evaluation</i>	
Ben Hachey, Joel Nothman and Will Radford	464
<i>Identifying Real-Life Complex Task Names with Task-Intrinsic Entities from Microblogs</i>	
Ting-Xuan Wang, Kun-Yu Tsai and Wen-Hsiang Lu	470
<i>Mutual Disambiguation for Entity Linking</i>	
Eric Charton, Marie-Jean Meurs, Ludovic Jean-Louis and Michel Gagnon	476
<i>How Well can We Learn Interpretable Entity Types from Text?</i>	
Dirk Hovy	482
<i>Learning Translational and Knowledge-based Similarities from Relevance Rankings for Cross-Language Retrieval</i>	
Shigehiko Schamoni, Felix Hieber, Artem Sokolov and Stefan Riezler	488
<i>Two-Stage Hashing for Fast Document Retrieval</i>	
Hao Li, Wei Liu and Heng Ji	495
<i>An Annotation Framework for Dense Event Ordering</i>	
Taylor Cassidy, Bill McDowell, Nathanael Chambers and Steven Bethard	501
<i>Linguistically debatable or just plain wrong?</i>	
Barbara Plank, Dirk Hovy and Anders Søgaard	507
<i>Humans Require Context to Infer Ironic Intent (so Computers Probably do, too)</i>	
Byron C. Wallace, Do Kook Choe, Laura Kertz and Eugene Charniak	512
<i>Automatic prediction of aspectual class of verbs in context</i>	
Annemarie Friedrich and Alexis Palmer	517
<i>Combining Word Patterns and Discourse Markers for Paradigmatic Relation Classification</i>	
Michael Roth and Sabine Schulte im Walde	524
<i>Applying a Naive Bayes Similarity Measure to Word Sense Disambiguation</i>	
Tong Wang and Graeme Hirst	531
<i>Fast Easy Unsupervised Domain Adaptation with Marginalized Structured Dropout</i>	
Yi Yang and Jacob Eisenstein	538
<i>Improving Lexical Embeddings with Semantic Knowledge</i>	
Mo Yu and Mark Dredze	545

<i>Optimizing Segmentation Strategies for Simultaneous Speech Translation</i>	
Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda and Satoshi Nakamura	551
<i>A joint inference of deep case analysis and zero subject generation for Japanese-to-English statistical machine translation</i>	
Taku Kudo, Hiroshi Ichikawa and Hideto Kazawa	557
<i>A Hybrid Approach to Skeleton-based Translation</i>	
Tong Xiao, Jingbo Zhu and Chunliang Zhang	563
<i>Effective Selection of Translation Model Training Data</i>	
Le Liu, Yu Hong, Hao Liu, Xing Wang and Jianmin Yao	569
<i>Refinements to Interactive Translation Prediction Based on Search Graphs</i>	
Philipp Koehn, Chara Tsoukala and Herve Saint-Amand	574
<i>Cross-lingual Model Transfer Using Feature Representation Projection</i>	
Mikhail Kozhevnikov and Ivan Titov	579
<i>Cross-language and Cross-encyclopedia Article Linking Using Mixed-language Topic Model and Hyponym Translation</i>	
Yu-Chun Wang, Chun-Kai Wu and Richard Tzong-Han Tsai	586
<i>Nonparametric Method for Data-driven Image Captioning</i>	
Rebecca Mason and Eugene Charniak	592
<i>Improved Correction Detection in Revised ESL Sentences</i>	
Huichao Xue and Rebecca Hwa	599
<i>Unsupervised Alignment of Privacy Policies using Hidden Markov Models</i>	
Rohan Ramanath, Fei Liu, Norman Sadeh and Noah A. Smith	605
<i>Enriching Cold Start Personalized Language Model Using Social Network Information</i>	
Yu-Yang Huang, Rui Yan, Tsung-Ting Kuo and Shou-De Lin	611
<i>Automatic Labelling of Topic Models Learned from Twitter by Summarisation</i>	
Amparo Elizabeth Cano Basave, Yulan He and Ruifeng Xu	618
<i>Stochastic Contextual Edit Distance and Probabilistic FSTs</i>	
Ryan Cotterell, Nanyun Peng and Jason Eisner	625
<i>Labelling Topics using Unsupervised Graph-based Methods</i>	
Nikolaos Aletras and Mark Stevenson	631
<i>Training a Korean SRL System with Rich Morphological Features</i>	
Young-Bum Kim, Heemoon Chae, Benjamin Snyder and Yu-Seop Kim	637
<i>Semantic Parsing for Single-Relation Question Answering</i>	
Wen-tau Yih, Xiaodong He and Christopher Meek	643
<i>On WordNet Semantic Classes and Dependency Parsing</i>	
Kepa Bengoetxea, Eneko Agirre, Joakim Nivre, Yue Zhang and Koldo Gojenola	649
<i>Enforcing Structural Diversity in Cube-pruned Dependency Parsing</i>	
Hao Zhang and Ryan McDonald	656

<i>The Penn Parsed Corpus of Modern British English: First Parsing Results and Analysis</i> Seth Kulick, Anthony Kroch and Beatrice Santorini	662
<i>Parser Evaluation Using Derivation Trees: A Complement to evalb</i> Seth Kulick, Ann Bies, Justin Mott, Anthony Kroch, Beatrice Santorini and Mark Liberman ..	668
<i>Learning Polylingual Topic Models from Code-Switched Social Media Documents</i> Nanyun Peng, Yiming Wang and Mark Dredze	674
<i>Normalizing tweets with edit scripts and recurrent neural embeddings</i> Grzegorz Chrupała	680
<i>Exponential Reservoir Sampling for Streaming Language Models</i> Miles Osborne, Ashwin Lall and Benjamin Van Durme	687
<i>A Piece of My Mind: A Sentiment Analysis Approach for Online Dispute Detection</i> Lu Wang and Claire Cardie	693
<i>A Simple Bayesian Modelling Approach to Event Extraction from Twitter</i> Deyu Zhou, Liangyu Chen and Yulan He	700
<i>Be Appropriate and Funny: Automatic Entity Morph Encoding</i> Boliang Zhang, Hongzhao Huang, Xiaoman Pan, Heng Ji, Kevin Knight, Zhen Wen, Yizhou Sun, Jiawei Han and Bulent Yener	706
<i>Applying Grammar Induction to Text Mining</i> Andrew Salway and Samia Touileb	712
<i>Semantic Consistency: A Local Subspace Based Method for Distant Supervised Relation Extraction</i> Xianpei Han and Le Sun	718
<i>Concreteness and Subjectivity as Dimensions of Lexical Meaning</i> Felix Hill and Anna Korhonen	725
<i>Infusion of Labeled Data into Distant Supervision for Relation Extraction</i> Maria Pershina, Bonan Min, Wei Xu and Ralph Grishman	732
<i>Recognizing Implied Predicate-Argument Relationships in Textual Inference</i> Asher Stern and Ido Dagan	739
<i>Measuring metaphoricity</i> Jonathan Dunn	745
<i>Empirical Study of Unsupervised Chinese Word Segmentation Methods for SMT on Large-scale Corpora</i> Xiaolin Wang, Masao Utiyama, Andrew Finch and Eiichiro Sumita	752
<i>EM Decipherment for Large Vocabularies</i> Malte Nuhn and Hermann Ney	759
<i>XMEANT: Better semantic MT evaluation without reference translations</i> Chi-kiu Lo, Meriem Beloucif, Markus Saers and Dekai Wu	765
<i>Sentence Level Dialect Identification for Machine Translation System Selection</i> Wael Salloum, Heba Elfardy, Linda Alamir-Salloum, Nizar Habash and Mona Diab	772

<i>RNN-based Derivation Structure Prediction for SMT</i> Feifei Zhai, Jiajun Zhang, Yu Zhou and Chengqing Zong	779
<i>Hierarchical MT Training using Max-Violation Perceptron</i> Kai Zhao, Liang Huang, Haitao Mi and Abe Ittycheriah	785
<i>Punctuation Processing for Projective Dependency Parsing</i> Ji Ma, Yue Zhang and Jingbo Zhu	791
<i>Transforming trees into hedges and parsing with "hedgebank" grammars</i> Mahsa Yarmohammadi, Aaron Dunlop and Brian Roark	797
<i>Incremental Predictive Parsing with TurboParser</i> Arne Köhn and Wolfgang Menzel	803
<i>Tailoring Continuous Word Representations for Dependency Parsing</i> Mohit Bansal, Kevin Gimpel and Karen Livescu	809
<i>Observational Initialization of Type-Supervised Taggers</i> Hui Zhang and John DeNero	816
<i>How much do word embeddings encode about syntax?</i> Jacob Andreas and Dan Klein	822
<i>Distributed Representations of Geographically Situated Language</i> David Bamman, Chris Dyer and Noah A. Smith	828
<i>Improving Multi-Modal Representations Using Image Dispersion: Why Less is Sometimes More</i> Douwe Kiela, Felix Hill, Anna Korhonen and Stephen Clark	835
<i>Bilingual Event Extraction: a Case Study on Trigger Type Determination</i> Zhu Zhu, Shoushan Li, Guodong Zhou and Rui Xia	842
<i>Understanding Relation Temporality of Entities</i> Taesung Lee and Seung-won Hwang	848
<i>Does the Phonology of L1 Show Up in L2 Texts?</i> Garrett Nicolai and Grzegorz Kondrak	854
<i>Cross-lingual Opinion Analysis via Negative Transfer Detection</i> Lin Gui, Ruifeng Xu, Qin Lu, Jun Xu, Jian Xu, Bin Liu and Xiaolong Wang	860

Conference Program

Sunday, June 22, 2014

7:30–18:00 Registration

7:30–9:00 Breakfast

9:00–12:30 Morning Tutorial

Session T1: Gaussian Processes for Natural Language Processing

Session T2: Scalable Large-Margin Structured Learning: Theory and Algorithms

Session T3: Semantics for Large-Scale Multimedia: New Challenges for NLP

Session T4: Wikification and Beyond: The Challenges of Entity and Concept Grounding

12:30–14:00 Lunch break

14:00–17:30 Afternoon Tutorial

Session T5: New Directions in Vector Space Models of Meaning

Session T6: Structured Belief Propagation for NLP

Session T7: Semantics, Discourse and Statistical Machine Translation

Session T8: Syntactic Processing Using Global Discriminative Learning and Beam-Search Decoding

18:00–21:00 Welcome Reception

Monday, June 23, 2014

7:30–18:00 Registration

7:30–9:00 Breakfast

8:55–9:00 Opening session

9:00–9:40 President talk

9:40–10:10 Coffee break

Session 1A: Discourse, Dialogue, Coreference and Pragmatics

Session 1B: Semantics I

Session 1C: Machine Translation I

Session 1D: Syntax, Parsing, and Tagging I

Session 1E: NLP for the Web and Social Media I

11:50–13:20 Lunch break; Student Lunch

Monday, June 23, 2014 (continued)

Session 2A: Syntax, Parsing and Tagging II

Session 2B: Semantics II

Session 2C: Word Segmentation and POS Tagging

Session 2D: SRW

Session 2E: Sentiment Analysis I

15:00–15:30 Coffee break

Session 3A: Topic Modeling

Session 3B: Information Extraction I

Session 3C: Generation

Session 3D: Syntax, Parsing and Tagging III

Session 3E: Language Resources and Evaluation I

16:45–17:00 Break

17:00–18:00 Invited talk I: Corinna Cortes

Monday, June 23, 2014 (continued)

Oral Sessions for Student Research Workshop Posters

18:50–21:30 Poster and Dinner Session I: TACL Papers, Long Papers, Short Papers, Student Research Workshop; Demonstrations

Exploring the Relative Role of Bottom-up and Top-down Information in Phoneme Learning
Abdellah Fourtassi, Thomas Schatz, Balakrishnan Varadarajan and Emmanuel Dupoux

Biases in Predicting the Human Language Model
Alex B. Fine, Austin F. Frank, T. Florian Jaeger and Benjamin Van Durme

Probabilistic Labeling for Efficient Referential Grounding based on Collaborative Discourse
Changsong Liu, Lanbo She, Rui Fang and Joyce Y. Chai

A Composite Kernel Approach for Dialog Topic Tracking with Structured Domain Knowledge from Wikipedia
Seokhwan Kim, Rafael E. Banchs and Haizhou Li

An Extension of BLANC to System Mentions
Xiaoqiang Luo, Sameer Pradhan, Marta Recasens and Eduard Hovy

Scoring Coreference Partitions of Predicted Mentions: A Reference Implementation
Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng and Michael Strube

Measuring Sentiment Annotation Complexity of Text
Aditya Joshi, Abhijit Mishra, Nivvedan Senthamilselvan and Pushpak Bhattacharyya

Improving Citation Polarity Classification with Product Reviews
Charles Jochim and Hinrich Schütze

Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification
Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou and Ke Xu

Sprinkling Topics for Weakly Supervised Text Classification
Swapnil Hingmire and Sutanu Chakraborti

A Feature-Enriched Tree Kernel for Relation Extraction
Le Sun and Xianpei Han

Monday, June 23, 2014 (continued)

Employing Word Representations and Regularization for Domain Adaptation of Relation Extraction

Thien Huu Nguyen and Ralph Grishman

Graph Ranking for Collective Named Entity Disambiguation

Ayman Alhelbawy and Robert Gaizauskas

Descending-Path Convolution Kernel for Syntactic Structures

Chen Lin, Timothy Miller, Alvin Kho, Steven Bethard, Dmitriy Dligach, Sameer Pradhan and Guergana Savova

Entities' Sentiment Relevance

Zvi Ben-Ami, Ronen Feldman and Binyamin Rosenfeld

Automatic Detection of Multilingual Dictionaries on the Web

Gintare Grigonyte and Timothy Baldwin

Automatic Detection of Cognates Using Orthographic Alignment

Alina Maria Ciobanu and Liviu P. Dinu

Automatically constructing Wordnet Synsets

Khang Nhut Lam, Feras Al Tarouti and Jugal Kalita

Constructing a Turkish-English Parallel TreeBank

Olcay Taner Yıldız, Ercan Solak, Onur Görgün and Razieh Ehsani

Improved Typesetting Models for Historical OCR

Taylor Berg-Kirkpatrick and Dan Klein

Robust Logistic Regression using Shift Parameters

Julie Tibshirani and Christopher D. Manning

Faster Phrase-Based Decoding by Refining Feature State

Kenneth Heafield, Michael Kayser and Christopher D. Manning

Decoder Integration and Expected BLEU Training for Recurrent Neural Network Language Models

Michael Auli and Jianfeng Gao

Monday, June 23, 2014 (continued)

On the Elements of an Accurate Tree-to-String Machine Translation System

Graham Neubig and Kevin Duh

Simple extensions and POS Tags for a reparameterised IBM Model 2

Douwe Gelling and Trevor Cohn

Dependency-based Pre-ordering for Chinese-English Machine Translation

Jingsheng Cai, Masao Utiyama, Eiichiro Sumita and Yujie Zhang

Generalized Character-Level Spelling Error Correction

Noura Farra, Nadi Tomeh, Alla Rozovskaya and Nizar Habash

Improved Iterative Correction for Distant Spelling Errors

Sergey Gubanov, Irina Galinskaya and Alexey Baytin

Predicting Grammaticality on an Ordinal Scale

Michael Heilman, Aoife Cahill, Nitin Madnani, Melissa Lopez, Matthew Mulholland and Joel Tetreault

I'm a Belieber: Social Roles via Self-identification and Conceptual Attributes

Charley Beller, Rebecca Knowles, Craig Harman, Shane Bergsma, Margaret Mitchell and Benjamin Van Durme

Automatically Detecting Corresponding Edit-Turn-Pairs in Wikipedia

Johannes Daxenberger and Iryna Gurevych

Two Knives Cut Better Than One: Chinese Word Segmentation with Dual Decomposition

Mengqiu Wang, Rob Voigt and Christopher D. Manning

Effective Document-Level Features for Chinese Patent Word Segmentation

Si Li and Nianwen Xue

Word Segmentation of Informal Arabic with Domain Adaptation

Will Monroe, Spence Green and Christopher D. Manning

Resolving Lexical Ambiguity in Tensor Regression Models of Meaning

Dimitri Kartsaklis, Nal Kalchbrenner and Mehrnoosh Sadrzadeh

Monday, June 23, 2014 (continued)

A Novel Content Enriching Model for Microblog Using News Corpus

Yunlun Yang, Zhihong Deng and Hongliang Yu

Learning Bilingual Word Representations by Marginalizing Alignments

Tomáš Kočiský, Karl Moritz Hermann and Phil Blunsom

Detecting Retries of Voice Search Queries

Rivka Levitan and David Elson

Sliding Alignment Windows for Real-Time Crowd Captioning

Mohammad Kazemi, Rahman Lavaee, Iftekhar Naim and Daniel Gildea

Detection of Topic and its Extrinsic Evaluation Through Multi-Document Summarization

Yoshimi Suzuki and Fumiyo Fukumoto

Content Importance Models for Scoring Writing From Sources

Beata Beigman Klebanov, Nitin Madnani, Jill Burstein and Swapna Somasundaran

Chinese Morphological Analysis with Character-level POS Tagging

Mo Shen, Hongxiao Liu, Daisuke Kawahara and Sadao Kurohashi

Part-of-Speech Tagging using Conditional Random Fields: Exploiting Sub-Label Dependencies for Improved Accuracy

Miikka Silfverberg, Teemu Ruokolainen, Krister Lindén and Mikko Kurimo

POS induction with distributional and morphological information using a distance-dependent Chinese restaurant process

Kairit Sirts, Jacob Eisenstein, Micha Elsner and Sharon Goldwater

Improving the Recognizability of Syntactic Relations Using Contextualized Examples

Aditi Muralidharan and Marti A. Hearst

Tuesday, June 24, 2014

7:30–18:00 Registration

7:30–9:00 Breakfast

9:00–10:00 Invited talk II: Zoran Popović

10:00–10:30 Coffee break

Session 4A: Machine Learning for NLP

Session 4B: Information Extraction II

Session 4C: Machine Translation II

Session 4D: Summarization

Session 4E: Language Resources and Evaluation II

12:10–13:30 Lunch break

Session 5A: Question Answering

Session 5B: Information Extraction III

Tuesday, June 24, 2014 (continued)

Session 5C: Lexical Semantics and Ontology I

Session 5D: Syntax, Parsing and Tagging IV

Session 5E: Cognitive Modeling and Psycholinguistics

14:45–15:15 Coffee break

Session 6A: Machine Translation III

15:15–15:30 *How to Speak a Language without Knowing It*
Xing Shi, Kevin Knight and Heng Ji

15:30–15:45 *Assessing the Discourse Factors that Influence the Quality of Machine Translation*
Junyi Jessy Li, Marine Carpuat and Ani Nenkova

15:45–16:00 *Automatic Detection of Machine Translated Text and Translation Quality Estimation*
Roei Aharoni, Moshe Koppel and Yoav Goldberg

Session 6B: Lexical Semantics and Ontology II

15:15–15:30 *Improving sparse word similarity models with asymmetric measures*
Jean Mark Gawron

15:30–15:45 *Dependency-Based Word Embeddings*
Omer Levy and Yoav Goldberg

15:45–16:00 *Vector spaces for historical linguistics: Using distributional semantics to study syntactic productivity in diachrony*
Florent Perek

Tuesday, June 24, 2014 (continued)

Session 6C: Generation/Summarization/Dialogue

- 15:15–15:30 *Single Document Summarization based on Nested Tree Structure*
Yuta Kikuchi, Tsutomu Hirao, Hiroya Takamura, Manabu Okumura and Masaaki Nagata
- 15:30–15:45 *Linguistic Considerations in Automatic Question Generation*
Karen Mazidi and Rodney D. Nielsen
- 15:45–16:00 *Polynomial Time Joint Structural Inference for Sentence Compression*
Xian Qian and Yang Liu
- 16:00–16:15 *A Bayesian Method to Incorporate Background Knowledge during Automatic Text Summarization*
Annie Louis
- 16:15–16:30 *Predicting Power Relations between Participants in Written Dialog from a Single Thread*
Vinodkumar Prabhakaran and Owen Rambow

Session 6D: NLP Applications and NLP Enabled Technology I

- 15:15–15:30 *Tri-Training for Authorship Attribution with Limited Training Data*
Tieyun Qian, Bing Liu, Li Chen and Zhiyong Peng
- 15:30–15:45 *Automation and Evaluation of the Keyword Method for Second Language Learning*
Gözde Özbal, Daniele Pighin and Carlo Strapparava
- 15:45–16:00 *Citation Resolution: A method for evaluating context-based citation recommendation systems*
Daniel Duma and Ewan Klein
- 16:00–16:15 *Hippocratic Abbreviation Expansion*
Brian Roark and Richard Sproat
- 16:15–16:30 *Unsupervised Feature Learning for Visual Sign Language Identification*
Binyam Gebrekidan Gebre, Onno Crasborn, Peter Wittenburg, Sebastian Drude and Tom Heskes

Tuesday, June 24, 2014 (continued)

Session 6E: Language Resources and Evaluation III

- 15:15–15:30 *Experiments with crowdsourced re-annotation of a POS tagging data set*
Dirk Hovy, Barbara Plank and Anders Søgaard
- 15:30–15:45 *Building Sentiment Lexicons for All Major Languages*
Yanqing Chen and Steven Skiena
- 15:45–16:00 *Difficult Cases: From Data to Learning, and Back*
Beata Beigman Klebanov and Eyal Beigman
- 16:00–16:15 *The VerbCorner Project: Findings from Phase 1 of crowd-sourcing a semantic decomposition of verbs*
Joshua K. Hartshorne, Claire Bonial and Martha Palmer
- 16:15–16:30 *A Corpus of Sentence-level Revisions in Academic Writing: A Step towards Understanding Statement Strength in Communication*
Chenhao Tan and Lillian Lee
- 16:50–19:20 Poster and Dinner Session II: Long Papers, Short Papers and Demonstrations in Grand Ballroom I/II/III/IV/V/VI/VII/VIII
- Determiner-Established Deixis to Communicative Artifacts in Pedagogical Text*
Shomir Wilson and Jon Oberlander
- Modeling Factuality Judgments in Social Media Text*
Sandeep Soni, Tanushree Mitra, Eric Gilbert and Jacob Eisenstein
- A Topic Model for Building Fine-grained Domain-specific Emotion Lexicon*
Min Yang, Dingju Zhu and Kam-Pui Chow
- Depeche Mood: a Lexicon for Emotion Analysis from Crowd Annotated News*
Jacopo Staiano and Marco Guerini
- Improving Twitter Sentiment Analysis with Topic-Based Mixture Modeling and Semi-Supervised Training*
Bing Xiang and Liang Zhou
- Cross-cultural Deception Detection*
Verónica Pérez-Rosas and Rada Mihalcea

Tuesday, June 24, 2014 (continued)

Particle Filter Rejuvenation and Latent Dirichlet Allocation

Chandler May, Alex Clemmer and Benjamin Van Durme

Comparing Automatic Evaluation Measures for Image Description

Desmond Elliott and Frank Keller

Learning a Lexical Simplifier Using Wikipedia

Colby Horn, Cathryn Manduca and David Kauchak

Cheap and easy entity evaluation

Ben Hachey, Joel Nothman and Will Radford

Identifying Real-Life Complex Task Names with Task-Intrinsic Entities from Microblogs

Ting-Xuan Wang, Kun-Yu Tsai and Wen-Hsiang Lu

Mutual Disambiguation for Entity Linking

Eric Charton, Marie-Jean Meurs, Ludovic Jean-Louis and Michel Gagnon

How Well can We Learn Interpretable Entity Types from Text?

Dirk Hovy

Learning Translational and Knowledge-based Similarities from Relevance Rankings for Cross-Language Retrieval

Shigehiko Schamoni, Felix Hieber, Artem Sokolov and Stefan Riezler

Two-Stage Hashing for Fast Document Retrieval

Hao Li, Wei Liu and Heng Ji

An Annotation Framework for Dense Event Ordering

Taylor Cassidy, Bill McDowell, Nathanael Chambers and Steven Bethard

Linguistically debatable or just plain wrong?

Barbara Plank, Dirk Hovy and Anders Søgaard

Humans Require Context to Infer Ironic Intent (so Computers Probably do, too)

Byron C. Wallace, Do Kook Choe, Laura Kertz and Eugene Charniak

Tuesday, June 24, 2014 (continued)

Automatic prediction of aspectual class of verbs in context

Annemarie Friedrich and Alexis Palmer

Combining Word Patterns and Discourse Markers for Paradigmatic Relation Classification

Michael Roth and Sabine Schulte im Walde

Applying a Naive Bayes Similarity Measure to Word Sense Disambiguation

Tong Wang and Graeme Hirst

Fast Easy Unsupervised Domain Adaptation with Marginalized Structured Dropout

Yi Yang and Jacob Eisenstein

Improving Lexical Embeddings with Semantic Knowledge

Mo Yu and Mark Dredze

Optimizing Segmentation Strategies for Simultaneous Speech Translation

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda and Satoshi Nakamura

A joint inference of deep case analysis and zero subject generation for Japanese-to-English statistical machine translation

Taku Kudo, Hiroshi Ichikawa and Hideto Kazawa

A Hybrid Approach to Skeleton-based Translation

Tong Xiao, Jingbo Zhu and Chunliang Zhang

Effective Selection of Translation Model Training Data

Le Liu, Yu Hong, Hao Liu, Xing Wang and Jianmin Yao

Refinements to Interactive Translation Prediction Based on Search Graphs

Philipp Koehn, Chara Tsoukala and Herve Saint-Amand

Cross-lingual Model Transfer Using Feature Representation Projection

Mikhail Kozhevnikov and Ivan Titov

Cross-language and Cross-encyclopedia Article Linking Using Mixed-language Topic Model and Hypernym Translation

Yu-Chun Wang, Chun-Kai Wu and Richard Tzong-Han Tsai

Tuesday, June 24, 2014 (continued)

Nonparametric Method for Data-driven Image Captioning

Rebecca Mason and Eugene Charniak

Improved Correction Detection in Revised ESL Sentences

Huichao Xue and Rebecca Hwa

Unsupervised Alignment of Privacy Policies using Hidden Markov Models

Rohan Ramanath, Fei Liu, Norman Sadeh and Noah A. Smith

Enriching Cold Start Personalized Language Model Using Social Network Information

Yu-Yang Huang, Rui Yan, Tsung-Ting Kuo and Shou-De Lin

Automatic Labelling of Topic Models Learned from Twitter by Summarisation

Amparo Elizabeth Cano Basave, Yulan He and Ruifeng Xu

Stochastic Contextual Edit Distance and Probabilistic FSTs

Ryan Cotterell, Nanyun Peng and Jason Eisner

Labelling Topics using Unsupervised Graph-based Methods

Nikolaos Aletras and Mark Stevenson

Training a Korean SRL System with Rich Morphological Features

Young-Bum Kim, Heemoon Chae, Benjamin Snyder and Yu-Seop Kim

Semantic Parsing for Single-Relation Question Answering

Wen-tau Yih, Xiaodong He and Christopher Meek

On WordNet Semantic Classes and Dependency Parsing

Kepa Bengoetxea, Eneko Agirre, Joakim Nivre, Yue Zhang and Koldo Gojenola

Enforcing Structural Diversity in Cube-pruned Dependency Parsing

Hao Zhang and Ryan McDonald

The Penn Parsed Corpus of Modern British English: First Parsing Results and Analysis

Seth Kulick, Anthony Kroch and Beatrice Santorini

Tuesday, June 24, 2014 (continued)

Parser Evaluation Using Derivation Trees: A Complement to evalb

Seth Kulick, Ann Bies, Justin Mott, Anthony Kroch, Beatrice Santorini and Mark Liberman

19:30–22:00 Social at the National Aquarium in Baltimore

Wednesday, June 25, 2014

7:30–18:00 Registration

7:30–9:00 Breakfast

Best paper session

10:15–10:45 Coffee break

Session 7A: Multimodal NLP/ Lexical Semantics

Session 7B: Semantics III

Session 7C: Machine Translation IV

Session 7D: NLP Applications and NLP Enabled Technology II

Session 7E: Sentiment Analysis II

12:25–13:30 Lunch break

13:30–15:00 ACL Business Meeting

Wednesday, June 25, 2014 (continued)

Session 8A: NLP for the Web and Social Media II

- 15:00–15:15 *Learning Polylingual Topic Models from Code-Switched Social Media Documents*
Nanyun Peng, Yiming Wang and Mark Dredze
- 15:15–15:30 *Normalizing tweets with edit scripts and recurrent neural embeddings*
Grzegorz Chrupała
- 15:30–15:45 *Exponential Reservoir Sampling for Streaming Language Models*
Miles Osborne, Ashwin Lall and Benjamin Van Durme
- 15:45–16:00 *A Piece of My Mind: A Sentiment Analysis Approach for Online Dispute Detection*
Lu Wang and Claire Cardie
- 16:00–16:15 *A Simple Bayesian Modelling Approach to Event Extraction from Twitter*
Deyu Zhou, Liangyu Chen and Yulan He
- 16:15–16:30 *Be Appropriate and Funny: Automatic Entity Morph Encoding*
Boliang Zhang, Hongzhao Huang, Xiaoman Pan, Heng Ji, Kevin Knight, Zhen Wen, Yizhou Sun, Jiawei Han and Bulent Yener

Session 8B: Semantics/Information Extraction

- 15:00–15:15 *Applying Grammar Induction to Text Mining*
Andrew Salway and Samia Touileb
- 15:15–15:30 *Semantic Consistency: A Local Subspace Based Method for Distant Supervised Relation Extraction*
Xianpei Han and Le Sun
- 15:30–15:45 *Concreteness and Subjectivity as Dimensions of Lexical Meaning*
Felix Hill and Anna Korhonen
- 15:45–16:00 *Infusion of Labeled Data into Distant Supervision for Relation Extraction*
Maria Pershina, Bonan Min, Wei Xu and Ralph Grishman
- 16:00–16:15 *Recognizing Implied Predicate-Argument Relationships in Textual Inference*
Asher Stern and Ido Dagan

Wednesday, June 25, 2014 (continued)

16:15–16:30 *Measuring metaphoricity*
Jonathan Dunn

Session 8C: Machine Translation V

15:00–15:15 *Empirical Study of Unsupervised Chinese Word Segmentation Methods for SMT on Large-scale Corpora*
Xiaolin Wang, Masao Utiyama, Andrew Finch and Eiichiro Sumita

15:15–15:30 *EM Decipherment for Large Vocabularies*
Malte Nuhn and Hermann Ney

15:30–15:45 *XMEANT: Better semantic MT evaluation without reference translations*
Chi-kiu Lo, Meriem Beloucif, Markus Saers and Dekai Wu

15:45–16:00 *Sentence Level Dialect Identification for Machine Translation System Selection*
Wael Salloum, Heba Elfardy, Linda Alamir-Salloum, Nizar Habash and Mona Diab

16:00–16:15 *RNN-based Derivation Structure Prediction for SMT*
Feifei Zhai, Jiajun Zhang, Yu Zhou and Chengqing Zong

16:15–16:30 *Hierarchical MT Training using Max-Violation Perceptron*
Kai Zhao, Liang Huang, Haitao Mi and Abe Ittycheriah

Session 8D: Syntax, Parsing, and Tagging V

15:00–15:15 *Punctuation Processing for Projective Dependency Parsing*
Ji Ma, Yue Zhang and Jingbo Zhu

15:15–15:30 *Transforming trees into hedges and parsing with "hedgebank" grammars*
Mahsa Yarmohammadi, Aaron Dunlop and Brian Roark

15:30–15:45 *Incremental Predictive Parsing with TurboParser*
Arne Köhn and Wolfgang Menzel

15:45–16:00 *Tailoring Continuous Word Representations for Dependency Parsing*
Mohit Bansal, Kevin Gimpel and Karen Livescu

Wednesday, June 25, 2014 (continued)

16:00–16:15 *Observational Initialization of Type-Supervised Taggers*
Hui Zhang and John DeNero

16:15–16:30 *How much do word embeddings encode about syntax?*
Jacob Andreas and Dan Klein

Session 8E: Multilinguality and Multimodal NLP

15:00–15:15 *Distributed Representations of Geographically Situated Language*
David Bamman, Chris Dyer and Noah A. Smith

15:15–15:30 *Improving Multi-Modal Representations Using Image Dispersion: Why Less is Sometimes More*
Douwe Kiela, Felix Hill, Anna Korhonen and Stephen Clark

15:30–15:45 *Bilingual Event Extraction: a Case Study on Trigger Type Determination*
Zhu Zhu, Shoushan Li, Guodong Zhou and Rui Xia

15:45–16:00 *Understanding Relation Temporality of Entities*
Taesung Lee and Seung-won Hwang

16:00–16:15 *Does the Phonology of L1 Show Up in L2 Texts?*
Garrett Nicolai and Grzegorz Kondrak

16:15–16:30 *Cross-lingual Opinion Analysis via Negative Transfer Detection*
Lin Gui, Ruifeng Xu, Qin Lu, Jun Xu, Jian Xu, Bin Liu and Xiaolong Wang

16:30–17:00 Coffee break

17:00–18:30 Lifetime Achievement Award

18:30–19:00 Closing session

Exploring the Relative Role of Bottom-up and Top-down Information in Phoneme Learning

Abdellah Fourtassi¹, Thomas Schatz^{1,2}, Balakrishnan Varadarajan³, Emmanuel Dupoux¹

¹ Laboratoire de Sciences Cognitives et Psycholinguistique, ENS/EHESS/CNRS, Paris, France

² SIERRA Project-Team, INRIA/ENS/CNRS, Paris, France

³ Center for Language and Speech Processing, JHU, Baltimore, USA

{abdellah.fourtassi; emmanuel.dupoux; balaji.iitm1}@gmail

thomas.schatz@laposte.net

Abstract

We test both bottom-up and top-down approaches in learning the phonemic status of the sounds of English and Japanese. We used large corpora of spontaneous speech to provide the learner with an input that models both the linguistic properties and statistical regularities of each language. We found both approaches to help discriminate between allophonic and phonemic contrasts with a high degree of accuracy, although top-down cues proved to be effective only on an interesting subset of the data.

1 Introduction

Developmental studies have shown that, during their first year, infants tune in on the phonemic categories (consonants and vowels) of their language, i.e., they lose the ability to distinguish some within-category contrasts (Werker and Tees, 1984) and enhance their ability to distinguish between-category contrasts (Kuhl et al., 2006). Current work in early language acquisition has proposed two competing hypotheses that purport to account for the acquisition of phonemes. The *bottom-up hypothesis* holds that infants converge on the linguistic units of their language through a similarity-based distributional analysis of their input (Maye et al., 2002; Vallabha et al., 2007). In contrast, the *top-down hypothesis* emphasizes the role of higher level linguistic structures in order to learn the lower level units (Feldman et al., 2013; Martin et al., 2013). The aim of the present work is to explore how much information can ideally be derived from both hypotheses.

The paper is organized as follows. First we describe how we modeled phonetic variation from audio recordings, second we introduce a bottom-up cue based on acoustic similarity and top-down cues based on the properties of the lexicon.

We test their performance in a task that consists in discriminating within-category contrasts from between-category contrasts. Finally we discuss the role and scope of each cue for the acquisition of phonemes.

2 Modeling phonetic variation

In this section, we describe how we modeled the representation of speech sounds putatively processed by infants, before they learn the relevant phonemic categories of their language. Following Peperkamp et al. (2006), we make the assumption that this input is quantized into context-dependent phone-sized unit we call *allophones*. Consider the example of the allophonic rule that applies to the French /r/:

$$/r/ \rightarrow \begin{cases} [\chi] / & \text{before a voiceless obstruent} \\ [ʀ] & \text{elsewhere} \end{cases}$$

Figure 1: Allophonic variation of French /r/

The phoneme /r/ surfaces as voiced ([ʀ]) before a voiced obstruent like in [kanaʀ ʒon] (“canard jaune”, yellow duck) and as voiceless ([χ]) before a voiceless obstruent as in [kanaχ puʁpʁ] (“canard pourpre”, purple duck). Assuming speech sounds are coded as allophones, the challenge facing the learner is to distinguish the allophonic variation ([ʀ], [χ]) from the phonemic variation (related to a difference in the meaning) like the contrast ([ʀ], [l]).

Previous work has generated allophonic variation using random contexts (Martin et al., 2013). This procedure does not take into account the fact that contexts belong to natural classes. In addition, it does not enable to compute an acoustic distance. Here, we generate linguistically and acoustically controlled allophones using Hidden Markov Models (HMMs) trained on audio recordings.

2.1 Corpora

We use two speech corpora: the Buckeye Speech corpus (Pitt et al., 2007), which consists of 40 hours of spontaneous conversations with 40 speakers of American English, and the core of the Corpus of Spontaneous Japanese (Maekawa et al., 2000) which also consists of about 40 hours of recorded spontaneous conversations and public speeches in different fields. Both corpora are time-aligned with phonetic labels. Following Boruta (2012), we relabeled the Japanese corpus using 25 phonemes. For English, we used the phonemic version which consists of 45 phonemes.

2.2 Input generation

2.2.1 HMM-based allophones

In order to generate linguistically and acoustically plausible allophones, we apply a standard Hidden Markov Model (HMM) phoneme recognizer with a three-state per phone architecture to the signal, as follows.

First, we convert the raw speech waveform of the corpora into successive vectors of Mel Frequency Cepstrum Coefficients (MFCC), computed over 25 ms windows, using a period of 10 ms (the windows overlap). We use 12 MFCC coefficients, plus the energy, plus the first and second order derivatives, yielding 39 dimensions per frame. Second, we start HMM training using one three-state model per phoneme. Third, each phoneme model is cloned into context-dependent triphone models, for each context in which the phoneme actually occurs (for example, the phoneme /a/ occurs in the context [d-a-g] as in the word /dag/ (“dog”). The triphone models are then retrained on only the relevant subset of the data, corresponding to the given triphone context. These detailed models are clustered back into inventories of various sizes (from 2 to 20 times the size of the phonemic inventory) using a linguistic feature-based decision tree, and the HMM states of linguistically similar triphones are tied together so as to maximize the likelihood of the data. Finally, the triphone models are trained again while the initial gaussian emission models are replaced by mixture of Gaussians with a progressively increasing number of components, until each HMM state is modeled by a mixture of 17 diagonal-covariance Gaussians. The HMM were built using the HMM Toolkit (HTK: Young et al., 2006).

2.2.2 Random allophones

As a control, we also reproduce the random allophones of Martin et al. (2013), in which allophonic contexts are determined randomly: for a given phoneme /p/, the set of all possible contexts is randomly partitioned into a fixed number n of subsets. In the transcription, the phoneme /p/ is converted into one of its allophones (p_1, p_2, \dots, p_n) depending on the subset to which the current context belongs.

3 Bottom-up and top-down hypotheses

3.1 Acoustic cue

The bottom-up cue is based on the hypothesis that instances of the same phoneme are likely to be acoustically more similar than instances of two different phonemes (see Cristia and Seidl, in press) for a similar proposition). In order to provide a proxy for the perceptual distance between allophones, we measure the information theoretic distance between the acoustic HMMs of these allophones. The 3-state HMMs of the two allophones were aligned with Dynamic Time Warping (DTW), using as a distance between pairs of emitting states, a symmetrized version of the Kullback-Leibler (KL) divergence measure (each state was approximated by a single non-diagonal Gaussian):

$$A(x, y) = \sum_{(i,j) \in DTW(x,y)} KL(N_{x_i} || N_{y_j}) + KL(N_{y_j} || N_{x_i})$$

Where $\{(i, j) \in DTW(x, y)\}$ is the set of index pairs over the HMM states that correspond to the optimal DTW path in the comparison between phone model x and y , and N_{x_i} the full covariance Gaussian distribution for state i of phone x . For obvious reasons, the acoustic distance cue cannot be computed for Random allophones.

3.2 Lexical cues

The top-down information we use in this study, is based on the insight of Martin et al. (2013). It rests on the idea that true lexical minimal pairs are not very frequent in human languages, as compared to minimal pairs due to mere phonological processes. In fact, the latter creates variants (alternants) of the same lexical item since adjacent sounds condition the realization of the first and final phoneme. For example, as shown in figure 1, the phoneme /r/ surfaces as [ɹ] or [ʁ] depending on whether or not the

next sound is a voiceless obstruent. Therefore, the lexical item /kanar/ surfaces as [kanaχ] or [kanaʁ]. The lexical cue assumes that a pair of words differing in the first or last segment (like [kanaχ] and [kanaʁ]) is more likely to be the result of a phonological process triggered by adjacent sounds, than a true semantic minimal pair.

However, this strategy clearly gives rise to false alarms in the (albeit relatively rare) case of true minimal pairs like [kanaχ] (“duck”) and [kanaʎ] (“canal”), where ([χ], [ʎ]) will be mistakenly labeled as allophonic.

In order to mitigate the problem of false alarms, we also use Boruta (2011)’s continuous version, where each pair of phones is characterized by the number of lexical minimal pairs it forms.

$$B(x, y) = |(Ax, Ay) \in L^2| + |(xA, yA) \in L^2|$$

where $\{Ax \in L\}$ is the set of words in the lexicon L that end in the phone x , and $\{(Ax, Ay) \in L^2\}$ is the set of phonological minimal pairs in $L \times L$ that vary on the final segment.

In addition, we introduce another cue that could be seen as a normalization of Boruta’s cue:

$$N(x, y) = \frac{|(Ax, Ay) \in L^2| + |(xA, yA) \in L^2|}{|\{Ax \in L\}| + |\{Ay \in L\}| + |\{xA \in L\}| + |\{yA \in L\}|}$$

4 Experiment

4.1 Task

For each corpus we list all the possible pairs of attested allophones. Some of these pairs are allophones of the same phoneme (allophonic pair) and others are allophones of different phonemes (non-allophonic pairs). The task is a same-different classification, whereby each of these pairs is given a score from the cue that is being tested. A good cue gives higher scores to allophonic pairs.

4.2 Evaluation

We use the same evaluation procedure as in Martin et al. (2013). It is carried out by computing the area under the curve of the Receiver Operating Characteristic (ROC). A value of 0.5 represents chance and a value of 1 represents perfect performance.

In order to lessen the potential influence of the structure of the corpus (mainly the order of the utterances) on the results, we use a statistical resampling scheme. The corpus is divided into small blocks (of 20 utterances each). In each run, we draw randomly with replacement from this set of

blocks a sample of the same size as the original corpus. This sample is then used to retrain the acoustic models and generate a phonetic inventory that we use to re-transcribe the corpus and re-compute the cues. We report scores averaged over 5 such runs.

4.3 Results

Table 1 shows the classification scores for the lexical cues when we vary the inventory size from 2 allophones per phoneme in average, to 20 allophones per phoneme, using the Random allophones. The top-down scores are very high, replicating Martin et al.’s results, and even improving the performance using Boruta’s cue and our new Normalized cue.

Allo./phon.	English			Japanese		
	M	B	N	M	B	N
2	0.784	0.935	0.951	0.580	0.989	1.00
5	0.845	0.974	0.982	0.653	0.978	0.991
10	0.886	0.974	0.981	0.733	0.944	0.971
20	0.918	0.961	0.966	0.785	0.869	0.886

Table 1 : Same-different scores for top-down cues on Random allophones, as a function of the average number of allophones per phoneme. M=Martin et al., B=Boruta, N=Normalized

Table 2 shows the results for HMM-based allophones. The acoustic score is very accurate for both languages and is quite robust to variation. Top-down cues, on the other hand, perform, surprisingly, almost at chance level in distinguishing between allophonic and non-allophonic pairs. A similar discrepancy for the case of Japanese was actually noted, but not explained, in Boruta (2012).

Allo./phon.	English				Japanese			
	A	M	B	N	A	M	B	N
2	0.916	0.592	0.632	0.643	0.885	0.422	0.524	0.537
5	0.918	0.592	0.607	0.611	0.908	0.507	0.542	0.551
10	0.893	0.569	0.571	0.571	0.827	0.533	0.546	0.548
20	0.879	0.560	0.560	0.559	0.876	0.541	0.543	0.543

Table 2 : Same-different scores for bottom-up and top-down cues on HMM-based allophones, as a function of the average number of allophones per phoneme. A=Acoustic, M=Martin et al., B=Boruta, N=Normalized

5 Analysis

5.1 Why does the performance drop for realistic allophones?

When we list all possible pairs of allophones in the inventory, some of them correspond to lexi-

cal alternants ([ç], [ʁ]) → ([kanaç] and [kanaʁ]), others to true minimal pairs ([ʁ], [l]) → ([kanaʁ] and [kanal]), and yet others will simply not generate lexical variation at all, we will call those: *invisible* pairs. For instance, in English, /h/ and /ɥ/ occur in different syllable positions and thus cannot appear in any minimal pair. As defined above, top-down cues are set to 0 in such pairs (which means that they are systematically classified as non-allophonic). This is a correct decision for /h/ vs. /ɥ/, but not for invisible pairs that also happen to be allophonic, resulting in false negatives. In tables 3, we show that, indeed, invisible pairs is a major issue, and could explain to a large extent the pattern of results found above. In fact, the proportion of visible allophonic pairs (“allo” column) is way lower for HMM-based allophones. This means that the majority of allophonic pairs in the HMM case are invisible, and therefore, will be mistakenly classified as non-allophonic.

	Random				HMM			
	English		Japanese		English		Japanese	
Allo./phon.	allo	¬ allo	allo	¬ allo	allo	¬ allo	allo	¬ allo
2	92.9	36.3	100	83.9	48.9	25.3	37.1	53.2
5	97.2	28.4	99.6	69.0	31.1	14.3	25.0	25.9
10	96.8	19.9	96.7	50.1	19.8	4.23	21.0	14.4
20	94.3	10.8	83.4	26.4	14.0	1.89	12.4	4.04

Table 3 : Proportion (in %) of allophonic pairs (allo), and non-allophonic pairs (¬ allo) associated with at least one lexical minimal pair, in Random and HMM allophones.

There are basically two reasons why an allophonic pair would be invisible (will not generate lexical alternants). The first one is the absence of evidence, e.g., if the edges of the word with the underlying phoneme do not appear in enough contexts to generate the corresponding variants. This happens when the corpus is so small that no word ending with, say, /r/ appears in both voiced and voiceless contexts. The second, is when the allophones are triggered on maximally different contexts (on the right and the left) as illustrated below:

$$/p/ \rightarrow \begin{cases} [p_1] / A_B \\ [p_2] / C_D \end{cases}$$

When A doesn’t overlap with C and B does not overlap with D, it becomes impossible for the pair ([p₁], [p₂]) to generate a lexical minimal pair. This is simply because a pair of allophones needs to share at least one context to be able to form variants of a word (the second or penultimate segment of this word).

When asked to split the set of contexts in two distinct categories that trigger [p₁] and [p₂] (i.e., A__B and C__D), the random procedure will often make A overlap with B and C overlap with D because it is completely oblivious to any acoustic or linguistic similarity, thus making it always possible for the pair of allophones to generate lexical alternants. A more realistic categorization (like the HMM-based one), will naturally tend to minimize within-category distance, and maximize between-category distance. Therefore, we will have less overlap, making the chances of the pair to generate a lexical pair smaller. The more allophones we have, the bigger is the chance to end up with non-overlapping categories (invisible allophonic pairs), and the more mistakes will be made, as shown in Table 3.

5.2 Restricting the role of top-down cues

The analysis above shows that top-down cues cannot be used to classify all contrasts. The approximation that consists in considering all pairs that do not generate lexical pairs as non-allophonic, does not scale up to realistic input. A more intuitive, but less ambitious, assumption is to restrict the scope of top-down cues to contrasts that do generate lexical variation (lexical alternants or true minimal pairs). Thus, they remain completely agnostic to the status of invisible pairs. This restriction makes sense since top-down information boils down to knowing whether two word forms belong to the same lexical category (reducing variation to allophony), or to two different categories (variation is then considered non-allophonic). Phonetic variation that does not cause lexical variation is, in this particular sense, orthogonal to our knowledge about the lexicon.

We test this hypothesis by applying the cues only to the subset of pairs that are associated with at least one lexical minimal pair. We vary the number of allophones per phoneme on the one hand (Table 4) and the size of the input on the other hand (Table 5). We refer to this subset by an asterisk (*), by which we also mark the cues that apply to it. Notice that, in this new framing, the M cue is completely uninformative since it assigns the same value to all pairs.

As predicted, the cues perform very well on this subset, especially the N cue. The combination of top-down and bottom-up cues shows that the former is always useful, and that these two sources of

Allo./phon.	English						Japanese							
	* (%)	Individual cues				Combination		* (%)	Individual cues				Combination	
		A	A*	B*	N*	A*+B*	A*+N*		A	A*	B*	N*	A*+B*	A*+N*
2	26.6	0.916	0.965	0.840	0.950	0.971	0.994	60.92	0.885	0.909	0.859	0.906	0.918	0.946
4	14.3	0.918	0.964	0.858	0.951	0.975	0.991	30.88	0.908	0.917	0.850	0.936	0.934	0.976
10	4.24	0.893	0.937	0.813	0.939	0.960	0.968	16.06	0.827	0.839	0.899	0.957	0.904	0.936
20	1.67	0.879	0.907	0.802	0.907	0.942	0.940	5.02	0.876	0.856	0.882	0.959	0.913	0.950

Table 4 : Same-different scores for different cues and their combinations with HMM-allophones, as a function of average number of allophones per phonemes.

Size (hours)	English						Japanese							
	* (%)	Individual cues				Combination		* (%)	Individual cues				Combination	
		A	A*	B*	N*	A*+B*	A*+N*		A	A*	B*	N*	A*+B*	A*+N*
1	9.87	0.885	0.907	0.741	0.915	0.927	0.969	34.78	0.890	0.883	0.835	0.915	0.889	0.934
4	18.3	0.918	0.958	0.798	0.917	0.967	0.989	48.00	0.917	0.939	0.860	0.937	0.938	0.973
8	21.3	0.916	0.964	0.837	0.942	0.971	0.992	51.71	0.915	0.940	0.889	0.937	0.954	0.977
20	24.4	0.911	0.960	0.827	0.936	0.969	0.994	58.12	0.921	0.954	0.865	0.912	0.945	0.971
40	26.6	0.916	0.965	0.840	0.950	0.971	0.994	60.92	0.885	0.909	0.859	0.906	0.918	0.946
∞	34.82							72.16						

Table 5 : Same-different scores for different cues and their combinations with HMM-allophones, as a function of corpus size.

* (%) refers to the proportion of the subset of contrasts associated with at least one minimal pair. The cues applied to this subset are marked with an asterisk (*)

information are not completely redundant. However, the scope of top-down cues (the proportion of the subset *) shrinks as we increase the number of allophones. Table 5 shows that this problem can, in principle, be mitigated by increasing the amount of data available to the learner. As we were limited to only 40 hours of speech, we generated an artificial corpus that uses the same lexicon but with all possible word orders so as to maximize the number of contexts in which words appear. This artificial corpus increases the proportion of the subset, but we are still not at 100 % coverage, which according the analysis above, is due (at least in part) to the irreducible set of non-overlapping pairs.

6 Conclusion

In this study we explored the role of both bottom-up and top-down hypotheses in learning the phonemic status of the sounds of two typologically different languages. We introduced a bottom-up cue based on acoustic similarity, and we used already existing top-down cues to which we provided a new extension. We tested these hypotheses on English and Japanese, providing the learner with an input that mirrors closely the linguistic and acoustic properties of each language. We showed, on the one hand, that the bottom-up cue is a very reliable source of information, across different levels of variation and even with small amount of data. Top-down cues, on the other hand, were found to be effective only on a subset of the data,

which corresponds to the interesting contrasts that cause lexical variation. Their role becomes more relevant as the learner gets more linguistic experience, and their combination with bottom-up cues shows that they can provide non-redundant information. Note, finally, that even if this work is based on a more realistic input compared to previous studies, it still uses simplifying assumptions, like ideal word segmentation, and no low-level acoustic variability. Those assumptions are, however, useful in quantifying the information that can ideally be extracted from the input, which is a necessary preliminary step before modeling *how* this input is used in a cognitively plausible way. Interested readers may refer to (Fourtassi and Dupoux, 2014; Fourtassi et al., 2014) for a more learning-oriented approach, where some of the assumptions made here about high level representations are relaxed.

Acknowledgments

This project is funded in part by the European Research Council (ERC-2011-AdG-295810 BOOTPHON), the Agence Nationale pour la Recherche (ANR-10-LABX-0087 IEC, ANR-10-IDEX-0001-02 PSL*), the Fondation de France, the Ecole de Neurosciences de Paris, and the Région Ile de France (DIM cerveau et pensée). We thank Luc Boruta, Sanjeev Khudanpur, Isabelle Dautriche, Sharon Peperkamp and Benoit Crabbé for highly useful discussions and contributions.

References

- Luc Boruta. 2011. Combining Indicators of Allophony. In *Proceedings ACL-SRW*, pages 88–93.
- Luc Boruta. 2012. *Indicateurs d'allophonie et de phonémicité*. Doctoral dissertation, Université Paris-Diderot - Paris VII.
- A. Cristia and A. Seidl. In press. The hyperarticulation hypothesis of infant-directed speech. *Journal of Child Language*.
- Naomi H. Feldman, Thomas L. Griffiths, Sharon Goldwater, and James L. Morgan. 2013. A role for the developing lexicon in phonetic category acquisition. *Psychological Review*, 120(4):751–778.
- Abdellah Fourtassi and Emmanuel Dupoux. 2014. A rudimentary lexicon and semantics help bootstrap phoneme acquisition. In *Proceedings of the 18th Conference on Computational Natural Language Learning (CoNLL)*.
- Abdellah Fourtassi, Ewan Dunbar, and Emmanuel Dupoux. 2014. Self-consistency as an inductive bias in early language acquisition. In *Proceedings of the 36th Annual Meeting of the Cognitive Science Society*.
- Patricia K. Kuhl, Erica Stevens, Akiko Hayashi, Toshisada Deguchi, Shigeru Kiritani, and Paul Iversen. 2006. Infants show a facilitation effect for native language phonetic perception between 6 and 12 months. *Developmental Science*, 9(2):F13–F21.
- Kikuo Maekawa, Hanae Koiso, Sadaoki Furui, and Hitoshi Isahara. 2000. Spontaneous speech corpus of Japanese. In *LREC*, pages 947–952, Athens, Greece.
- Andrew Martin, Sharon Peperkamp, and Emmanuel Dupoux. 2013. Learning phonemes with a protolexicon. *Cognitive Science*, 37(1):103–124.
- J. Maye, J. F. Werker, and L. Gerken. 2002. Infant sensitivity to distributional information can affect phonetic discrimination. *Cognition*, 82:B101–B111.
- Sharon Peperkamp, Rozenn Le Calvez, Jean-Pierre Nadal, and Emmanuel Dupoux. 2006. The acquisition of allophonic rules: Statistical learning with linguistic constraints. *Cognition*, 101(3):B31–B41.
- M. A. Pitt, L. Dilley, K. Johnson, S. Kiesling, W. Raymond, E. Hume, and Fosler-Lussier. 2007. Buckeye corpus of conversational speech.
- G.K. Vallabha, J.L. McClelland, F. Pons, J.F. Werker, and S. Amano. 2007. Unsupervised learning of vowel categories from infant-directed speech. *Proceedings of the National Academy of Sciences*, 104(33):13273.
- Janet F. Werker and Richard C. Tees. 1984. Cross-language speech perception: Evidence for perceptual reorganization during the first year of life. *Infant Behavior and Development*, 7(1):49 – 63.
- Steve J. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland. 2006. *The HTK Book Version 3.4*. Cambridge University Press.

Biases in Predicting the Human Language Model

Alex B. Fine

University of Illinois at Urbana-Champaign
abfine@illinois.edu

Austin F. Frank

Riot Games
aufrank@riotgames.com

T. Florian Jaeger

University of Rochester
fjaeger@bcs.rochester.edu

Benjamin Van Durme

Johns Hopkins University
vandurme@cs.jhu.edu

Abstract

We consider the prediction of three human behavioral measures – lexical decision, word naming, and picture naming – through the lens of domain bias in language modeling. Contrasting the predictive ability of statistics derived from 6 different corpora, we find intuitive results showing that, e.g., a British corpus overpredicts the speed with which an American will react to the words *ward* and *duke*, and that the Google n-grams overpredicts familiarity with technology terms. This study aims to provoke increased consideration of the human language model by NLP practitioners: biases are not limited to differences between corpora (i.e. “train” vs. “test”); they can exist as well between corpora and the intended user of the resultant technology.

1 Introduction

Computational linguists build statistical language models for aiding in natural language processing (NLP) tasks. Computational psycholinguists build such models to aid in their study of human language processing. Errors in NLP are measured with tools like precision and recall, while errors in psycholinguistics are defined as failures to model a target phenomenon.

In the current study, we exploit errors of the latter variety—failure of a language model to predict human performance—to investigate *bias* across several frequently used corpora in computational linguistics. The human data is revealing because it trades on the fact that human language processing is *probability-sensitive*: language processing

reflects implicit knowledge of probabilities computed over linguistic units (e.g., words). For example, the amount of time required to read a word varies as a function of how predictable that word is (McDonald and Shillcock, 2003). Thus, failure of a language model to predict human performance reveals a mismatch between the language model and the human language model, i.e., bias.

Psycholinguists have known for some time that the ability of a corpus to explain behavior depends on properties of the corpus and the subjects (cf. Balota et al. (2004)). We extend that line of work by directly analyzing and quantifying this bias, and by linking the results to methodological concerns in both NLP and psycholinguistics.

Specifically, we predict human data from three widely used psycholinguistic experimental paradigms—lexical decision, word naming, and picture naming—using unigram frequency estimates from Google n-grams (Brants and Franz, 2006), Switchboard (Godfrey et al., 1992), spoken and written English portions of CELEX (Baayen et al., 1995), and spoken and written portions of the British National Corpus (BNC Consortium, 2007). While we find comparable overall fits of the behavioral data from all corpora under consideration, our analyses also reveal specific domain biases. For example, Google n-grams overestimates the ease with which humans will process words related to the web (*tech*, *code*, *search*, *site*), while the Switchboard corpus—a collection of informal telephone conversations between strangers—overestimates how quickly humans will react to colloquialisms (*heck*, *darn*) and backchannels (*wow*, *right*).

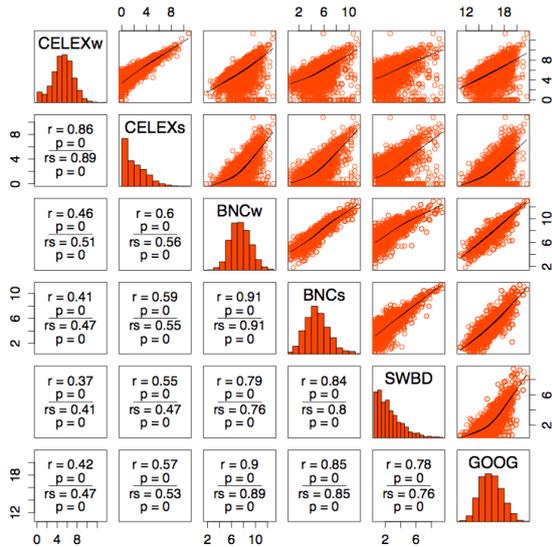


Figure 1: Pairwise correlations between log frequency estimates from each corpus. Histograms show distribution over frequency values from each corpus. Lower left panels give Pearson (top) and Spearman (bottom) correlation coefficients and associated p-values for each pair. Upper right panels plot correlations

2 Fitting Behavioral Data

2.1 Data

Pairwise Pearson correlation coefficients for log frequency were computed for all corpora under consideration. Significant correlations were found between log frequency estimates for all pairs (Figure 1). Intuitive biases are apparent in the correlations, e.g.: BNCw correlates heavily with BNCs (0.91), but less with SWBD (0.79), while BNCs correlates more with SWBD (0.84).¹

Corpus	Size (tokens)
Google n-grams (web release)	~ 1 trillion
British National Corpus (written, BNCw)	~ 90 million
British National Corpus (spoken, BNCs)	~ 10 million
CELEX (written, CELEXw)	~ 16.6 million
CELEX (spoken, CELEXs)	~ 1.3 million
Switchboard (Penn Treebank subset 3)	~ 800,000

Table 1: Summary of the corpora under consideration.

2.2 Approach

We ask whether domain biases manifest as systematic errors in predicting human behavior. Log unigram frequency estimates were derived from each corpus and used to predict reaction times (RTs) from three experiments employing *lexical*

¹BNCw and BNCs are both British, while BNCs and SWBD are both spoken.

decision (time required by subjects to correctly identify a string of letters as a word of English (Balota et al., 1999)); *word naming* (time required to read aloud a visually presented word (Spieler and Balota, 1997); (Balota and Spieler, 1998)); and *picture naming* (time required to say a picture’s name (Bates et al., 2003)). Previous work has shown that more frequent words lead to faster RTs. These three measures provide a strong test for the biases present in these corpora, as they span written and spoken lexical comprehension and production.

To compare the predictive strength of log frequency estimates from each corpus, we fit mixed effects regression models to the data from each experiment. As controls, all models included (1) mean log bigram frequency for each word, (2) word category (noun, verb, etc.), (3) log morphological family size (number of inflectional and derivational morphological family members), (4) number of synonyms, and (5) the first principal component of a host of orthographic and phonological features capturing neighborhood effects (type and token counts of orthographic and phonological neighbors as well as forward and backward inconsistent words; (Baayen et al., 2006)). Models of lexical decision and word naming included random intercepts of participant age to adjust for differences in mean RTs between old (mean age = 72) vs. young (mean age = 23) subjects, given differences between younger vs. older adults’ processing speed (cf. (Ramscar et al., 2014)). (All participants in the picture naming study were college students.)

2.3 Results

For each of the six panels corresponding to frequency estimates from a corpus A , Figure 2 gives the χ^2 value resulting from the log-likelihood ratio of (1) a model containing A and an estimate from one of the five remaining corpora (given on the x axis) and (2) a model containing just the corpus indicated on the x axis. Thus, for each panel, each bar in Figure 2 shows the explanatory power of estimates from the corpus given at the top of the panel after controlling for estimates from each of the other corpora.

Model fits reveal intuitive, previously undocumented biases in the ability of each corpus to predict human data. For example, corpora of British English tend to explain relatively little after con-

trolling for other British corpora in modeling lexical decision RTs (yellow). Similarly, Switchboard provides relatively little explanatory power over the other corpora in predicting picture naming RTs (blue bars), possibly because highly imageable nouns and verbs frequent in everyday interactions are underrepresented in telephone conversations between people with no common visual experience. In other words, idiosyncratic facts about the topics, dialects, etc. represented in each corpus lead to systematic patterns in how well each corpus can predict human data relative to the others. In some cases, the predictive value of one corpus after controlling for another—apparently for reasons related to genre, dialect—can be quite large (cf. the χ^2 difference between a model with both Google and Switchboard frequency estimates compared to one with only Switchboard [top right yellow bar]).

In addition to comparing the overall predictive power of the corpora, we examined the words for which behavioral predictions derived from the corpora deviated most from the observed behavior (word frequencies strongly over- or underestimated by each corpora). First, in Table 2 we give the ten words with the greatest relative difference in frequency for each corpus pair. For example, *fife* is deemed more frequent according to the BNC than to Google.²

These results suggest that particular corpora may be genre-biased in systematic ways. For instance, Google appears to be biased towards terminology dealing with adult material and technology. Similarly, BNCw is biased, relative to Google, towards Britishisms. For these words in the BNC and Google, we examined errors in predicted lexical decision times. Figure 3 plots errors in the linear model’s prediction of RTs for older (top) and younger (bottom) subjects.

The figure shows a positive correlation between how large the difference is between the lexical decision RT predicted by the model and the actually observed RT, and how over-estimated the log frequency of that word is in the BNC relative to Google (left panel) or in Google relative to the BNC (right panel). The left panel shows that BNC produces a much greater estimate of the log fre-

quency of the word *lee* relative to Google, which leads the model to predict a lower RT for this word than is observed (i.e., the error is positive; though note that the error is less severe for older relative to younger subjects). By contrast, the asymmetry between the two corpora in the estimated frequency of *sir* is less severe, so the observed RT deviates less from the predicted RT. In the right panel, we see that Google assigns a much greater estimate of log frequency to the word *tech* than the BNC, which leads a model predicting RTs from Google-derived frequency estimates to predict a far lower RT for this word than observed.

3 Discussion

Researchers in computational linguistics often assume that more data is always better than less data (Banko and Brill, 2001). This is true insofar as larger corpora allow computational linguists to generate *less noisy* estimates of the average language experience of the users of computational linguistics applications. However, corpus size does not necessarily eliminate certain types of *biases* in estimates of human linguistic experience, as demonstrated in Figure 3.

Our analyses reveal that 6 commonly used corpora fail to reflect the human language model in various ways related to dialect, modality, and other properties of each corpus. Our results point to a type of bias in commonly used language models that has been previously overlooked. This bias may limit the effectiveness of NLP algorithms intended to generalize to a linguistic domains whose statistical properties are generated by humans.

For psycholinguists these results support an important methodological point: while each corpus presents systematic biases in how well it predicts human behavior, all six corpora are, on the whole, of comparable predictive value and, specifically, the results suggest that the web performs as well as traditional instruments in predicting behavior. This has two implications for psycholinguistic research. First, as argued by researchers such as Lew (2009), given the size of the Web compared to other corpora, research focusing on low-frequency linguistic events—or requiring knowledge of the distributional characteristics of varied contexts—is now more tractable. Second, the viability of the web in predicting behavior opens up possibilities for computational psycholinguistic research in languages for which no corpora exist (i.e., most

²Surprisingly, *fife* was determined to be one of the words with the largest frequency asymmetry between Switchboard and the Google n-grams corpus. This was a result of lower-casing all of the words in the analyses, and the fact that Barney Fife was mentioned several times in the BNC.

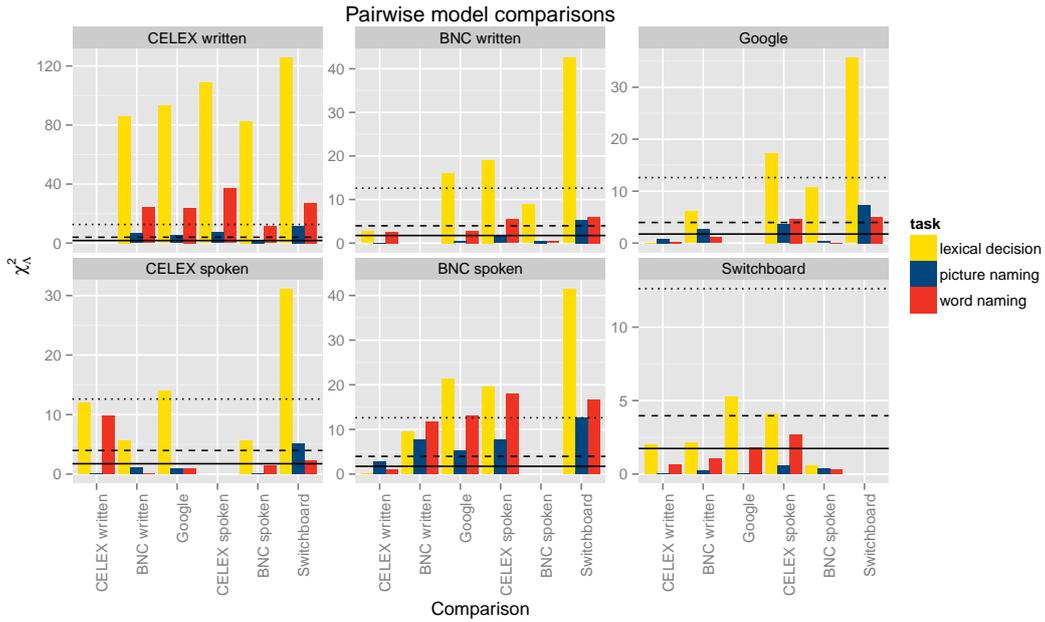


Figure 2: Results of log likelihood ratio model comparisons. Large values indicate that the reference predictor (panel title) explained a large amount of variance over and above the predictor given on the x-axis.

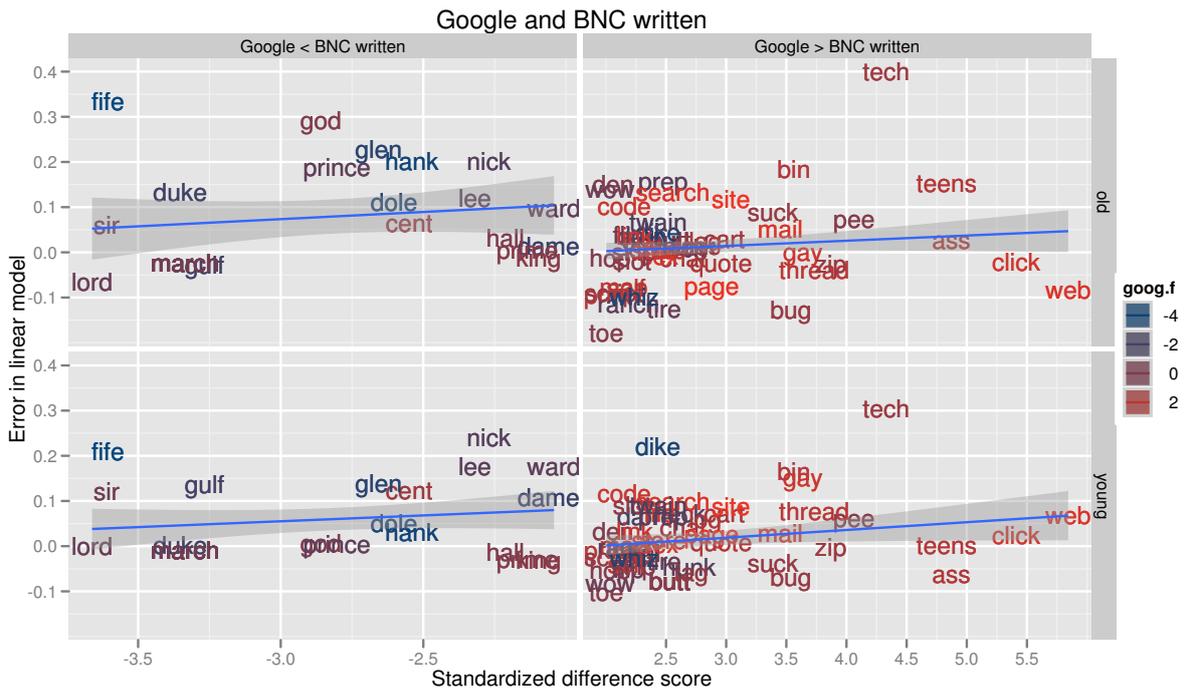


Figure 3: Errors in the linear model predicting lexical decision RTs from log frequency are plotted against the standardized difference in log frequency in the Google n-grams corpus versus the written portion of the BNC. Top and bottom panels show errors for older and younger subjects, respectively. The left panel plots words with much greater frequency in the written portion of the BNC relative to Google; the right panel plots words occurring more frequently in Google. Errors in the linear model are plotted against the standardized difference in log frequency across the corpora, and word color encodes the degree to which each word is more (red) or less (blue) frequent in Google. That the fit line in each graph is above 0 in the y-axis means that on average these biased words in each domain are being over-predicted, i.e., the corpus frequencies suggest humans will react (sometimes much) faster than they actually did in the lab.

Greater	Lesser	Top-10
google	bnc.s	web, ass, gay, tire, text, tool, code, woe, site, zip
google	bnc.w	ass, teens, tech, gay, bug, suck, site, cart, log, search
google	celex.s	teens, cart, gay, zip, mail, bin, tech, click, pee, site
google	celex.w	web, full, gay, bin, mail, zip, site, sake, ass, log
google	swbd	gay, thread, text, search, site, link, teens, seek, post, sex
bnc.w	google	fife, lord, duke, march, dole, god, cent, nick, dame, draught
bnc.w	bnc.s	pact, corps, foe, tract, hike, ridge, dine, crest, aide, whim
bnc.w	celex.s	staff, nick, full, waist, ham, lap, knit, sheer, bail, march
bnc.w	celex.w	staff, lord, last, nick, fair, glen, low, march, should, west
bnc.w	swbd	rose, prince, seek, cent, text, clause, keen, breach, soul, rise
celex.s	google	art, yes, pound, spoke, think, mean, say, thing, go, drove
celex.s	bnc.s	art, hike, pact, howl, ski, corps, peer, spoke, jazz, are
celex.s	bnc.w	art, yes, dike, think, thing, sort, mean, write, pound, lot
celex.s	celex.w	yes, sort, thank, think, jazz, heck, tape, well, fife, get
celex.s	swbd	art, cell, rose, spoke, aim, seek, shall, seed, text, knight
celex.w	google	art, plod, pound, shake, spoke, dine, howl, sit, say, draught
celex.w	bnc.s	hunch, stare, strife, hike, woe, aide, rout, yell, glaze, flee
celex.w	bnc.w	dike, whiz, dine, shake, grind, jerk, whoop, say, are, cram
celex.w	celex.s	wrist, pill, lawn, clutch, stare, spray, jar, shark, plead, horn
celex.w	swbd	art, rose, seek, aim, rise, burst, seed, cheek, grin, lip
swbd	google	mow, kind, lot, think, fife, corps, right, cook, sort, do
swbd	bnc.s	creek, mow, guess, pact, strife, tract, hank, howl, foe, nap
swbd	bnc.w	stuff, whiz, tech, lot, kind, creek, darn, dike, bet, kid
swbd	celex.s	wow, sauce, mall, deck, full, spray, flute, rib, guy, bunch
swbd	celex.w	heck, guess, right, full, stuff, lot, last, well, guy, fair

Table 2: Examples of words with largest difference in z-transformed log frequencies (e.g., the relative frequencies of *fife*, *lord*, and *duke*, in the BNC are far greater than in Google).

languages). This furthers the arguments of the “the web as corpus” community (Kilgarriff and Grefenstette, 2003) with respect to psycholinguistics.

Finally, combining multiple sources of frequency estimates is one way researchers may be able to reduce the prediction bias from any single corpus. This relates to work in automatically building domain specific corpora (e.g., Moore and Lewis (2010), Axelrod et al. (2011), Daumé III and Jagarlamudi (2011), Wang et al. (2014), Gao et al. (2002), and Lin et al. (1997)). Those efforts focus on building representative document collections for a target domain, usually based on a seed set of initial documents. Our results prompt the question: can one use human behavior as the *target* in the construction of such a corpus? Concretely, can we build corpora by optimizing an objective measure that minimizes error in predicting human reaction times? Prior work in building balanced corpora used either rough estimates of the ratio of genre styles a normal human is exposed to daily (e.g., the Brown corpus (Kucera and Francis, 1967)), or simply sampled text evenly across genres (e.g., COCA: the Corpus of Contemporary American English (Davies, 2009)). Just as language models have been used to predict reading grade-level of documents (Collins-Thompson and Callan, 2004), human language models could be

used to predict the appropriateness of a document for inclusion in an “automatically balanced” corpus.

4 Conclusion

We have shown intuitive, domain-specific biases in the prediction of human behavioral measures via corpora of various genres. While some psycholinguists have previously acknowledged that different corpora carry different predictive power, this is the first work to our knowledge to systematically document these biases across a range of corpora, and to relate these predictive errors to domain bias, a pressing issue in the NLP community. With these results in hand, future work may now consider the automatic construction of a “properly” balanced text collection, such as originally desired by the creators of the Brown corpus.

Acknowledgments

The authors wish to thank three anonymous ACL reviewers for helpful feedback. This research was supported by a DARPA award (FA8750-13-2-0017) and NSF grant IIS-0916599 to BVD, NSF IIS-1150028 CAREER Award and Alfred P. Sloan Fellowship to TFJ, and an NSF Graduate Research Fellowship to ABF.

References

- A. Axelrod, X. He, and J. Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 11)*.
- R. H. Baayen, R. Piepenbrock, and L. Gulikers. 1995. The CELEX Lexical Database (Release 2). Linguistic Data Consortium, Philadelphia.
- R. H. Baayen, L. F. Feldman, and R. Schreuder. 2006. Morphological influences on the recognition of monosyllabic monomorphemic words. *Journal of Memory and Language*, 53:496–512.
- D. A. Balota and D. H. Spieler. 1998. The utility of item-level analyses in model evaluation: A reply to Seidenberg & Plaut (1998). *Psychological Science*.
- D. A. Balota, M. J. Cortese, and M. Pilotti. 1999. Item-level analyses of lexical decision performance: Results from a mega-study. In *Abstracts of the 40th Annual Meeting of the Psychonomics Society*, page 44.
- D. Balota, M. Cortese, S. Sergent-Marshall, D. Spieler, and M. Yap. 2004. Visual word recognition for single-syllable words. *Journal of Experimental Psychology: General*, (133):283316.
- M. Banko and E. Brill. 2001. Mitigating the paucity of data problem. *Human Language Technology*.
- E. Bates, S. D’Amico, T. Jacobsen, A. Szkely, E. Andonova, A. Devescovi, D. Herron, CC Lu, T. Pechmann, C. Plh, N. Wicha, K. Federmeier, I. Gerdjikova, G. Gutierrez, D. Hung, J. Hsu, G. Iyer, K. Kohnert, T. Mehotcheva, A. Orozco-Figueroa, A. Tzeng, and O. Tzeng. 2003. Timed picture naming in seven languages. *Psychonomic Bulletin & Review*, 10(2):344–380.
- BNC Consortium. 2007. The British National Corpus, version 3 (BNC XML Edition). Distributed by Oxford University Computing Services on behalf of the BNC Consortium.
- T. Brants and A. Franz. 2006. Web 1T 5-gram Version 1. Linguistic Data Consortium (LDC).
- Kevyn Collins-Thompson and James P. Callan. 2004. A language modeling approach to predicting reading difficulty. In *HLT-NAACL*, pages 193–200.
- H. Daumé III and J. Jagarlamudi. 2011. Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 11)*.
- M. Davies. 2009. The 385+ million word corpus of contemporary american english (19902008+): Design, architecture, and linguistic insights. *International Journal of Corpus Linguistics*, 14(2):159–190.
- J. Gao, J. Goodman, M. Li, and K. F. Lee. 2002. Toward a unified approach to statistical language modeling for chinese. In *Proceedings of the ACM Transactions on Asian Language Information Processing (TALIP 02)*.
- J. Godfrey, E. Holliman, and J. McDaniel. 1992. SWITCHBOARD: Telephone Speech Corpus for Research and Development. In *Proceedings of ICASSP-92*, pages 517–520.
- A. Kilgarriff and G. Grefenstette. 2003. Introduction to the special issue on the web as corpus. *Computational Linguistics*, 29(3):333–348.
- H. Kucera and W.N. Francis. 1967. Computational analysis of present-day american english. providence, ri: Brown university press.
- R. Lew, 2009. *Contemporary Corpus Linguistics*, chapter The Web as corpus versus traditional corpora: Their relative utility for linguists and language learners, pages 289–300. London/New York: Continuum.
- S. C. Lin, C. L. Tsai, L. F. Chien, K. J. Chen, and L. S. Lee. 1997. Chinese language model adaptation based on document classification and multiple domain-specific language models. In *Proceedings of the 5th European Conference on Speech Communication and Technology*.
- S.A. McDonald and R.C. Shillcock. 2003. Eye movements reveal the on-line computation of lexical probabilities during reading. *Psychological science*, 14(6):648–52, November.
- R. C. Moore and W. Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 10)*.
- M. Ramscar, P. Hendrix, C. Shaoul, P. Milin, and R. H. Baayen. 2014. The myth of cognitive decline: non-linear dynamics of lifelong learning. *Topics in Cognitive Science*, 32:5–42.
- D. H. Spieler and D. A. Balota. 1997. Bringing computational models of word naming down to the item level. 6:411–416.
- L. Wang, D.F. Wong, L.S. Chao, Y. Lu, and J. Xing. 2014. A systematic comparison of data selection criteria for smt domain adaptation. *The Scientific World Journal*.

Probabilistic Labeling for Efficient Referential Grounding based on Collaborative Discourse

Changsong Liu, Lanbo She, Rui Fang, Joyce Y. Chai

Department of Computer Science and Engineering

Michigan State University

East Lansing, MI 48824

{cliu, shelanbo, fangrui, jchai}@cse.msu.edu

Abstract

When humans and artificial agents (e.g. robots) have mismatched perceptions of the shared environment, referential communication between them becomes difficult. To mediate perceptual differences, this paper presents a new approach using probabilistic labeling for referential grounding. This approach aims to integrate different types of evidence from the collaborative referential discourse into a unified scheme. Its probabilistic labeling procedure can generate multiple grounding hypotheses to facilitate follow-up dialogue. Our empirical results have shown the probabilistic labeling approach significantly outperforms a previous graph-matching approach for referential grounding.

1 Introduction

In situated human-robot dialogue, humans and robots have mismatched capabilities of perceiving the shared environment. Thus referential communication between them becomes extremely challenging. To address this problem, our previous work has conducted a simulation-based study to collect a set of human-human conversation data that explain how partners with mismatched perceptions strive to succeed in referential communication (Liu et al., 2012; Liu et al., 2013). Our data have shown that, when conversation partners have mismatched perceptions, they tend to make extra collaborative effort in referential communication. For example, the speaker often refers to the intended object iteratively: first issuing an initial *installment*, and then *refashioning* till the hearer identifies the referent correctly. The hearer, on the other hand, often provides useful feedback based on which further refashioning can be made.

This data has demonstrated the importance of incorporating collaborative discourse for referential grounding.

Based on this data, as a first step we developed a graph-matching approach for referential grounding (Liu et al., 2012; Liu et al., 2013). This approach uses *Attributed Relational Graph* to capture collaborative discourse and employs a state-space search algorithm to find proper grounding results. Although it has made meaningful progress in addressing collaborative referential grounding under mismatched perceptions, the state-space search based approach has two major limitations. First, it is neither flexible to obtain multiple grounding hypotheses, nor flexible to incorporate different hypotheses incrementally for follow-up grounding. Second, the search algorithm tends to have a high time complexity for optimal solutions. Thus, the previous approach is not ideal for collaborative and incremental dialogue systems that interact with human users in real time.

To address these limitations, this paper describes a new approach to referential grounding based on probabilistic labeling. This approach aims to integrate different types of evidence from the collaborative referential discourse into a unified probabilistic scheme. It is formulated under the Bayesian reasoning framework to easily support generation and incorporation of multiple grounding hypotheses for follow-up processes. Our empirical results have shown that the probabilistic labeling approach significantly outperforms the state-space search approach in both grounding accuracy and efficiency. This new approach provides a good basis for processing collaborative discourse and enabling collaborative dialogue system in situated referential communication.

2 Related Work

Previous works on situated referential grounding have mainly focused on computational models that connect linguistic referring expressions to the perceived environment (Gorniak and Roy, 2004; Gorniak and Roy, 2007; Siebert and Schlangen, 2008; Matuszek et al., 2012; Jayant and Thomas, 2013). These works have provided valuable insights on how to manually and/or automatically build key components (e.g., semantic parsing, grounding functions between visual features and words, mapping procedures) for a situated referential grounding system. However, most of these works only dealt with the interpretation of single referring expressions, rather than interrelated expressions in collaborative dialogue.

Some earlier work (Edmonds, 1994; Heeman and Hirst, 1995) proposed a symbolic reasoning (i.e. planning) based approach to incorporate collaborative dialogue. However, in situated settings pure symbolic approaches will not be sufficient and new approaches that are robust to uncertainties need to be pursued. DeVault and Stone (2009) proposed a hybrid approach which combined symbolic reasoning and machine learning for interpreting referential grounding dialogue. But their “environment” was a simplistic block world and the issue of mismatched perceptions was not addressed.

3 Data

Previously, we have collected a set of human-human dialogues on an object-naming task (Liu et al., 2012). To simulate mismatched perceptions between a human and an artificial agent, two participants were shown different versions of an image: the *director* was shown the original image containing some randomly placed objects (e.g., fruits), and the *matcher* was shown an impoverished version of the image generated by computer vision. They were instructed to communicate with each other to figure out the identities of some “named” objects (only known to the director), such that the matcher could also know which object has what name.

Here is an example excerpt from this dataset:

- D*¹: there is basically a cluster of four objects in the upper left, do you see that (1)
M: yes (2)
D: ok, so the one in the corner is a blue cup (3)

¹*D* stands for the *director*; *M* stands for the *matcher*.

- M*: I see there is a square, but fine, it is blue (4)
D: alright, I will just go with that, so and then right under that is a yellow pepper (5)
M: ok, I see apple but orangish yellow (6)
D: ok, so that yellow pepper is named Brittany (7)
M: uh, the bottom left of those four? Because I do see a yellow pepper in the upper right (8)
D: the upper right of the four of them? (9)
M: yes (10)
D: ok, so that is basically the one to the right of the blue cup (11)
M: yeah (12)
D: that is actually an apple (13)

As we can see from this example, both the director and the matcher make extra efforts to overcome the mismatched perceptions through collaborative dialogue. Our ultimate goal is to develop computational approaches that can ground interrelated referring expressions to the physical world, and enable collaborative actions of the dialogue agent (similar to the active role that the matcher played in the human-human dialogue). For the time being, we use this data to evaluate our computational approach for referential grounding, namely, replacing the matcher by our automatic system to ground the director’s referring expressions.

4 Probabilistic Labeling for Reference Grounding

4.1 System Overview

Our system first processes the data using automatic semantic parsing and coreference resolution. For semantic parsing, we use a rule-based CCG parser (Bozsahin et al., 2005) to parse each utterance into a formal semantic representation. For example, the utterance “a pear is to the right of the apple” is parsed as

$$[a_1, a_2], [Pear(a_1), Apple(a_2), RightOf(a_1, a_2)]$$

which consists of a list of *discourse entities* (e.g., a_1 and a_2) and a list of first-order-logic predicates that specify the unary attributes of these entities and the binary relations between them.

We then perform pairwise coreference resolution on the discourse entities to find out the discourse relations between entities from different utterances. Formally, let a_i be a discourse entity extracted from the current utterance, and a_j a discourse entity from a previous utterance. We train a maximum entropy classifier² (Manning and Klein,

²The features we use for the classification include the distance between a_i and a_j , the determiners associated with them, the associated pronouns, the syntactic roles, the extracted unary properties, etc.

2003) to predict whether a_i and a_j should refer to the same object (i.e. *positive*) or to different objects (i.e. *negative*).

Based on the semantic parsing and pairwise coreference resolution results, our system further builds a graph representation to capture the collaborative discourse and formulate referential grounding as a probabilistic labeling problem, as described next.

4.2 Graph Representation

We use an *Attributed Relational Graph* (Tsai and Fu, 1979) to represent the referential grounding discourse (which we call the “*dialogue graph*”). It is constructed based on the semantic parsing and coreference resolution results. The dialogue graph contains a set A of N nodes:

$$A = \{a_1, a_2, \dots, a_N\}$$

in which each node a_i represents a discourse entity from the parsing results. And for each pair of nodes a_i and a_j there can be an edge $a_i a_j$ that represents the physical or discourse relation (i.e. coreference) between the two nodes.

Furthermore, each node a_i can be assigned a set of “attributes”:

$$\mathbf{x}_i = \{x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(K)}\}$$

which are used to specify information about the unary properties of the corresponding discourse entity. Similarly, each edge $a_i a_j$ can also be assigned a set of attributes \mathbf{x}_{ij} to specify information about the binary relations between two discourse entities. The node attributes are from the semantic parsing results, i.e., the unary properties associated to a discourse entity. The edge attributes can be either from parsing results, such as a spatial relation between two entities (e.g., *RightOf*(a_1, a_2)); Or from pairwise coreference resolution results, i.e., two entities are coreferential (*coref* = +) or not (*coref* = -).

Besides the dialogue graph that represents the linguistic discourse, we build another graph to represent the perceived environment. This graph is called the “*vision graph*” (since this graph is built based on computer vision’s outputs). It has a set Ω of M nodes:

$$\Omega = \{\omega_1, \omega_2, \dots, \omega_M\}$$

in which each node ω_α represents a physical object in the scene. Similar to the dialogue graph,

the vision graph also has edges (e.g., $\omega_\alpha \omega_\beta$), node attributes (e.g., $\check{\mathbf{x}}_\alpha$) and edge attributes (e.g., $\check{\mathbf{x}}_{\alpha\beta}$). Note that the attributes in the vision graph mostly have numeric values extracted by computer vision algorithms, whereas the attributes in the dialogue graph have symbolic values extracted from the linguistic discourse. A set of “symbol grounding functions” are used to bridge between the heterogeneous attributes (described later).

Given these two graph representations, referential grounding then can be formulated as a “*node labeling*” process, that is to assign a label θ_i to each node a_i . The value of θ_i can be any of the M node labels from the set Ω .

4.3 Probabilistic Labeling Algorithm

The probabilistic labeling algorithm (Christmas et al., 1995) is formulated in the Bayesian framework. It provides a unified evidence-combining scheme to integrate unary attributes, binary relations and prior knowledge for updating the labeling probabilities (i.e. $P(\theta_i = \omega_\alpha)$). The algorithm finds proper labelings in an iterative manner: it first initiates the labeling probabilities by considering only the unary attributes of each node, and then updates the labeling probability of each node based on the labeling of its neighbors and the relations with them.

Initialization:

Compute the initial labeling probabilities:

$$P^{(0)}(\theta_i = \omega_\alpha) = \frac{P(a_i | \theta_i = \omega_\alpha) \hat{P}(\theta_i = \omega_\alpha)}{\sum_{\omega_\lambda \in \Omega} P(a_i | \theta_i = \omega_\lambda) \hat{P}(\theta_i = \omega_\lambda)}$$

in which $\hat{P}(\theta_i = \omega_\alpha)$ is the prior probability of labeling a_i with ω_α . The prior probability can be used to encode any prior knowledge about possible labelings. Especially in incremental processing of the dialogue, the prior can encode previous grounding hypotheses, and other information from the collaborative dialogue such as confirmation, rejection, or replacement.

$P(a_i | \theta_i = \omega_\alpha)$ is called the “compatibility coefficient” between a_i and ω_α , which is computed based on the attributes of a_i and ω_α :

$$\begin{aligned} P(a_i | \theta_i = \omega_\alpha) &= P(\mathbf{x}_i | \theta_i = \omega_\alpha) \\ &\approx \prod_k P(x_i^{(k)} | \theta_i = \omega_\alpha) \end{aligned}$$

and we further define

$$\begin{aligned}
P\left(x_i^{(k)} \mid \theta_i = \omega_\alpha\right) &= p\left(x_i^{(k)} \mid \check{x}_\alpha^{(k)}\right) \\
&= \frac{p\left(\check{x}_\alpha^{(k)} \mid x_i^{(k)}\right)p\left(x_i^{(k)}\right)}{\sum_{x_j^{(k)} \in L^{(k)}} p\left(\check{x}_\alpha^{(k)} \mid x_j^{(k)}\right)p\left(x_j^{(k)}\right)}
\end{aligned}$$

where $L^{(k)}$ is the ‘‘lexicon’’ for the k -th attribute of a dialogue graph node, e.g., for the *color* attribute:

$$L^{(k)} = \{red, green, blue, \dots\}$$

and $p\left(\check{x}_\alpha^{(k)} \mid x_i^{(k)}\right)$ is what we call a ‘‘symbol grounding function’’, i.e., the probability of observing $\check{x}_\alpha^{(k)}$ given the word $x_i^{(k)}$. It judges the compatibilities between the symbolic attribute values from the dialogue graph and the numeric attribute values from the vision graph. These symbol grounding functions can be either manually defined or automatically learned. In our current work, we use a set of manually defined grounding functions motivated by previous work (Gorniak and Roy, 2004).

Iteration:

Once the initial probabilities are calculated, the labeling procedure iterates till all the labeling probabilities have converged or the number of iterations has reached a specified limit. At each iteration and for each possible labeling, it computes a ‘‘support function’’ as:

$$\begin{aligned}
Q^{(n)}(\theta_i = \omega_\alpha) &= \prod_{j \in N_i} \sum_{\omega_\beta \in \Omega} P^{(n)}(\theta_j = \omega_\beta) \\
&P(a_i a_j \mid \theta_i = \omega_\alpha, \theta_j = \omega_\beta)
\end{aligned}$$

and updates the probability of each possible labeling as:

$$P^{(n+1)}(\theta_i = \omega_\alpha) = \frac{P^{(n)}(\theta_i = \omega_\alpha) Q^{(n)}(\theta_i = \omega_\alpha)}{\sum_{\omega_\lambda \in \Omega} P^{(n)}(\theta_i = \omega_\lambda) Q^{(n)}(\theta_i = \omega_\lambda)}$$

The support function $Q^{(n)}(\theta_i = \omega_\alpha)$ expresses how the labeling $\theta_i = \omega_\alpha$ at the n -th iteration is supported by the labeling of a_i ’s neighbors³, taking into consideration the binary relations that exist between a_i and them. Similar to the node compatibility coefficient, the edge compatibility coefficient between $a_i a_j$ and $\omega_\alpha \omega_\beta$,

³The set of indices N_i is defined as:

$$N_i = \{1, 2, \dots, i-1, i+1, \dots, N\}$$

	Top-1	Top-2	Top-3
Random Guess ^a	7.7%	15.4%	23.1%
S.S.S.	19.1%	19.7%	21.3%
P.L.	24.9%	36.1%	45.0%
Gain ^b	5.8% ($p < 0.01$)	16.4% ($p < 0.001$)	23.7% ($p < 0.001$)
P.L. using annotated coreference	66.4%	74.8%	81.9%

^aEach image contains an average of 13 objects.

^b p -value is based on the Wilcoxon signed-rank test (Wilcoxon et al., 1970) on the 62 dialogues.

Table 1: Comparison of the reference grounding performances of a random guess baseline, Probabilistic Labeling (P.L.) and State-Space Search (S.S.S.), and P.L. using manually annotated coreference.

namely the $P(a_i a_j \mid \theta_i = \omega_\alpha, \theta_j = \omega_\beta)$ for computing $Q^{(n)}(\theta_i = \omega_\alpha)$, is also based on the attributes of the two edges and their corresponding symbol grounding functions. So we also manually defined a set of grounding functions for edge attributes such as the spatial relation (e.g., *RightOf*, *Above*). If an edge is used to encode the discourse relation between two entities (i.e., the pairwise coreference results), the compatibility coefficient can be defined as (suppose edge $a_i a_j$ encodes a *positive* coreference relation between entities a_i and a_j):

$$\begin{aligned}
P(\overline{a_i a_j} = + \mid \theta_i = \omega_\alpha, \theta_j = \omega_\beta) \\
= \frac{P(\theta_i = \omega_\alpha, \theta_j = \omega_\beta \mid \overline{a_i a_j} = +) P(\overline{a_i a_j} = +)}{P(\theta_i = \omega_\alpha, \theta_j = \omega_\beta)}
\end{aligned}$$

which can be calculated based on the results from the coreference classifier (Section 4.1).

5 Evaluation and Discussion

Our dataset has 62 dialogues, each of which contains an average of 25 valid utterances from the director. We first applied the semantic parser and coreference classifier as described in Section 4.1 to process each dialogue, and then built a graph representation based on the automatic processing results at the end of the dialogue. On average, a dialogue graph consists of 33 discourse entities from the director’s utterances that need to be grounded.

We then applied both the probabilistic labeling algorithm and the state-space search algorithm to ground each of the director’s discourse entities onto an object perceived from the image. The averaged grounding accuracies of the two algorithms

are shown in the middle part of Table 1. The first column of Table 1 shows the grounding accuracies of the algorithm’s top-1 grounding hypothesis (i.e., $\theta_i = \underset{\omega_\alpha}{\operatorname{argmax}} P(\theta_i = \omega_\alpha)$ for each i). The second and third column then show the “accuracies” of the top-2 and top-3 hypotheses⁴, respectively.

As shown in Table 1, probabilistic labeling (i.e. P.L.) significantly outperforms state-space search (S.S.S.), especially with regard to producing meaningful multiple grounding hypotheses. The state-space search algorithm actually only results in multiple hypotheses for the overall matching, and it fails to produce multiple hypotheses for many individual discourse entities. Multiple grounding hypotheses can be very useful to generate responses such as clarification questions or nonverbal feedback (e.g. pointing, gazing). For example, if there are two competing hypotheses, the dialogue manager can utilize them to generate a response like “I see two objects there, are you talking about this one (pointing to) or that one (pointing to the other)?”. Such proactive feedback is often an effective way in referential communication (Clark and Wilkes-Gibbs, 1986; Liu et al., 2013).

The probabilistic labeling algorithm not only produces better grounding results, it also runs much faster (with a running-time complexity of $O(MN^2)$,⁵ comparing to $O(N^4)$ of the state-space search algorithm⁶). Figure 1 shows the averaged running time of the state-space search algorithm on a Intel Core i7 1.60GHz CPU with 16G RAM computer (the running time of the probabilistic labeling algorithm is not shown in Figure 1 since it always takes less than 1 second to run). As we can see, when the size of the dialogue graph becomes greater than 15, state-space search takes more than 1 minute to run. The efficiency of the probabilistic labeling algorithm thus makes it more appealing for real-time interaction applications.

Although probabilistic labeling significantly outperforms the state-space search, the grounding performance is still rather poor (less than 50%)

⁴The accuracy of the top-2/top-3 grounding hypotheses is measured by whether the ground-truth reference is included in the top-2/top-3 hypotheses.

⁵ M is the number of nodes in the vision graph and N is the number of nodes in the dialogue graph.

⁶Beam search algorithm is applied to reduce the exponential $O(M^N)$ to $O(N^4)$.

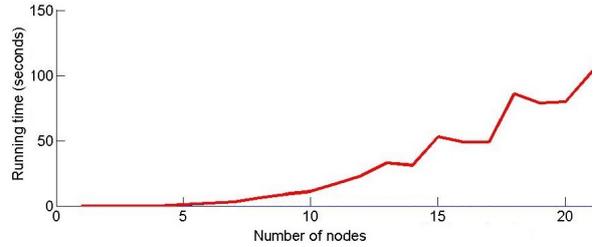


Figure 1: Average running time of the state-space search algorithm with respect to the number of nodes to be grounded in a dialogue graph.

even for the top-3 hypotheses. With no surprise, the coreference resolution performance plays an important role in the final grounding performance (see the grounding performance of using manually annotated coreference in the bottom part of Table 1). Due to the simplicity of our current coreference classifier and the flexibility of the human-human dialogue in the data, the pairwise coreference resolution only achieves 0.74 in precision and 0.43 in recall. The low recall of coreference resolution makes it difficult to link interrelated referring expressions and resolve them jointly. So it is important to develop more sophisticated coreference resolution and dialogue management components to reliably track the discourse relations and other dynamics in the dialogue to facilitate referential grounding.

6 Conclusion

In this paper, we have presented a probabilistic labeling based approach for referential grounding in situated dialogue. This approach provides a unified scheme for incorporating different sources of information. Its probabilistic scheme allows each information source to present multiple hypotheses to better handle uncertainties. Based on the integrated information, the labeling procedure then efficiently generates probabilistic grounding hypotheses, which can serve as important guidance for the dialogue manager’s decision making. In future work, we will utilize probabilistic labeling to incorporate information from verbal and nonverbal communication incrementally as the dialogue unfolds, and to enable collaborative dialogue agents in the physical world.

Acknowledgments

This work was supported by N00014-11-1-0410 from the Office of Naval Research and IIS-1208390 from the National Science Foundation.

References

- Cem Bozsahin, Geert-Jan M Kruijff, and Michael White. 2005. Specifying grammars for openccg: A rough guide. *Included in the OpenCCG distribution*.
- William J. Christmas, Josef Kittler, and Maria Petrou. 1995. Structural matching in computer vision using probabilistic relaxation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(8):749–764.
- Herbert H Clark and Deanna Wilkes-Gibbs. 1986. Referring as a collaborative process. *Cognition*, 22(1):1–39.
- David DeVault and Matthew Stone. 2009. Learning to interpret utterances using dialogue history. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 184–192. Association for Computational Linguistics.
- Philip G Edmonds. 1994. Collaboration on reference to objects that are not mutually known. In *Proceedings of the 15th conference on Computational linguistics-Volume 2*, pages 1118–1122. Association for Computational Linguistics.
- Peter Gorniak and Deb Roy. 2004. Grounded semantic composition for visual scenes. *J. Artif. Intell. Res.(JAIR)*, 21:429–470.
- Peter Gorniak and Deb Roy. 2007. Situated language understanding as filtering perceived affordances. *Cognitive Science*, 31(2):197–231.
- Peter A Heeman and Graeme Hirst. 1995. Collaborating on referring expressions. *Computational Linguistics*, 21(3):351–382.
- Krishnamurthy Jayant and Kollar Thomas. 2013. Jointly learning to parse and perceive: Connecting natural language to the physical world. *Transactions of the Association of Computational Linguistics*, 1:193–206.
- Changsong Liu, Rui Fang, and Joyce Chai. 2012. Towards mediating shared perceptual basis in situated dialogue. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 140–149, Seoul, South Korea, July. Association for Computational Linguistics.
- Changsong Liu, Rui Fang, Lanbo She, and Joyce Chai. 2013. Modeling collaborative referring for situated referential grounding. In *Proceedings of the SIG-DIAL 2013 Conference*, pages 78–86, Metz, France, August. Association for Computational Linguistics.
- Christopher Manning and Dan Klein. 2003. Optimization, maxent models, and conditional estimation without magic. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Tutorials - Volume 5*, NAACL-Tutorials '03, pages 8–8, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Cynthia Matuszek, Nicholas FitzGerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. 2012. A joint model of language and perception for grounded attribute learning. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ICML '12, pages 1671–1678, New York, NY, USA, July. Omnipress.
- Alexander Siebert and David Schlangen. 2008. A simple method for resolution of definite reference in a shared visual context. In *Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue*, pages 84–87. Association for Computational Linguistics.
- Wen-Hsiang Tsai and King-Sun Fu. 1979. Error-correcting isomorphisms of attributed relational graphs for pattern analysis. *Systems, Man and Cybernetics, IEEE Transactions on*, 9(12):757–768.
- Frank Wilcoxon, SK Katti, and Roberta A Wilcox. 1970. Critical values and probability levels for the wilcoxon rank sum test and the wilcoxon signed rank test. *Selected tables in mathematical statistics*, 1:171–259.

A Composite Kernel Approach for Dialog Topic Tracking with Structured Domain Knowledge from Wikipedia

Seokhwan Kim, Rafael E. Banchs, Haizhou Li

Human Language Technology Department

Institute for Infocomm Research

Singapore 138632

{kims, rembanchs, hli}@i2r.a-star.edu.sg

Abstract

Dialog topic tracking aims at analyzing and maintaining topic transitions in on-going dialogs. This paper proposes a composite kernel approach for dialog topic tracking to utilize various types of domain knowledge obtained from Wikipedia. Two kernels are defined based on history sequences and context trees constructed based on the extracted features. The experimental results show that our composite kernel approach can significantly improve the performances of topic tracking in mixed-initiative human-human dialogs.

1 Introduction

Human communications in real world situations interlace multiple topics which are related to each other in conversational contexts. This fact suggests that a dialog system should be also capable of conducting multi-topic conversations with users to provide them a more natural interaction with the system. However, the majority of previous work on dialog interfaces has focused on dealing with only a single target task. Although some multi-task dialog systems have been proposed (Lin et al., 1999; Ikeda et al., 2008; Celikyilmaz et al., 2011), they have aimed at just choosing the most probable one for each input from the sub-systems, each of which is independently operated from others.

To analyze and maintain dialog topics from a more systematic perspective in a given dialog flow, some researchers (Nakata et al., 2002; Lagus and Kuusisto, 2002; Adams and Martell, 2008) have considered this dialog topic identification as a separate sub-problem of dialog management and attempted to solve it with text categorization approaches for the recognized utterances in a given turn. The major obstacle to the success of these approaches results from the differences between

written texts and spoken utterances. In most text categorization tasks, the proper category for each textual unit can be assigned based only on its own content. However, the dialog topic at each turn can be determined not only by the user's intentions captured from the given utterances, but also by the system's decisions for dialog management purposes. Thus, the text categorization approaches can only be effective for the user-initiative cases when users tend to mention the topic-related expressions explicitly in their utterances.

The other direction of dialog topic tracking approaches made use of external knowledge sources including domain models (Roy and Subramaniam, 2006), heuristics (Young et al., 2007), and agendas (Bohus and Rudnicky, 2003; Lee et al., 2008). These knowledge-based methods have an advantage of dealing with system-initiative dialogs, because dialog flows can be controlled by the system based on given resources. However, this aspect can limit the flexibility to handle the user's responses which are contradictory to the system's suggestions. Moreover, these approaches face cost problems for building a sufficient amount of resources to cover broad states of complex dialogs, because these resources should be manually prepared by human experts for each specific domain.

In this paper, we propose a composite kernel to explore various types of information obtained from Wikipedia for mixed-initiative dialog topic tracking without significant costs for building resources. Composite kernels have been successfully applied to improve the performances in other NLP problems (Zhao and Grishman, 2005; Zhang et al., 2006) by integrating multiple individual kernels, which aim to overcome the errors occurring at one level by information from other levels. Our composite kernel consists of a history sequence and a domain context tree kernels, both of which are composed based on similar textual units in Wikipedia articles to a given dialog context.

t	Speaker	Utterance	Topic Transition
0	Guide	How can I help you?	NONE→NONE
1	Tourist	Can you recommend some good places to visit in Singapore?	NONE→ATTR
	Guide	Well if you like to visit an icon of Singapore, Merlion park will be a nice place to visit.	
2	Tourist	Merlion is a symbol for Singapore, right?	ATTR→ATTR
	Guide	Yes, we use that to symbolise Singapore.	
3	Tourist	Okay.	ATTR→ATTR
	Guide	The lion head symbolised the founding of the island and the fish body just symbolised the humble fishing village.	
4	Tourist	How can I get there from Orchard Road?	ATTR→TRSP
	Guide	You can take the north-south line train from Orchard Road and stop at Raffles Place station.	
5	Tourist	Is this walking distance from the station to the destination?	TRSP→TRSP
	Guide	Yes, it'll take only ten minutes on foot.	
6	Tourist	Alright.	TRSP→FOOD
	Guide	Well, you can also enjoy some seafoods at the riverside near the place.	
7	Tourist	What food do you have any recommendations to try there?	FOOD→FOOD
	Guide	If you like spicy foods, you must try chilli crab which is one of our favourite dishes here in Singapore.	
8	Tourist	Great! I'll try that.	FOOD→FOOD

Figure 1: Examples of dialog topic tracking on Singapore tour guide dialogs

2 Dialog Topic Tracking

Dialog topic tracking can be considered as a classification problem to detect topic transitions. The most probable pair of topics at just before and after each turn is predicted by the following classifier: $f(x_t) = (y_{t-1}, y_t)$, where x_t contains the input features obtained at a turn t , $y_t \in C$, and C is a closed set of topic categories. If a topic transition occurs at t , y_t should be different from y_{t-1} . Otherwise, both y_t and y_{t-1} have the same value.

Figure 1 shows an example of dialog topic tracking in a given dialog fragment on Singapore tour guide domain between a tourist and a guide. This conversation is divided into three segments, since f detects three topic transitions at t_1 , t_4 and t_6 . Then, a topic sequence of ‘Attraction’, ‘Transportation’, and ‘Food’ is obtained from the results.

3 Wikipedia-based Composite Kernel for Dialog Topic Tracking

The classifier f can be built on the training examples annotated with topic labels using supervised machine learning techniques. Although some fundamental features extracted from the utterances mentioned at a given turn or in a certain number of previous turns can be used for training the model, this information obtained solely from an ongoing dialog is not sufficient to identify not only user-initiative, but also system-initiative topic transitions.

To overcome this limitation, we propose to leverage on Wikipedia as an external knowledge source that can be obtained without significant

effort toward building resources for topic tracking. Recently, some researchers (Wilcock, 2012; Breuing et al., 2011) have shown the feasibility of using Wikipedia knowledge to build dialog systems. While each of these studies mainly focuses only on a single type of information including category relatedness or hyperlink connectedness, this work aims at incorporating various knowledge obtained from Wikipedia into the model using a composite kernel method.

Our composite kernel consists of two different kernels: a history sequence kernel and a domain context tree kernel. Both represent the current dialog context at a given turn with a set of relevant Wikipedia paragraphs which are selected based on the cosine similarity between the term vectors of the recently mentioned utterances and each paragraph in the Wikipedia collection as follows:

$$\text{sim}(x, p_i) = \frac{\phi(x) \cdot \phi(p_i)}{|\phi(x)| |\phi(p_i)|},$$

where x is the input, p_i is the i -th paragraph in the Wikipedia collection, $\phi(p_i)$ is the term vector extracted from p_i . The term vector for the input x , $\phi(x)$, is computed by accumulating the weights in the previous turns as follows:

$$\phi(x) = (\alpha_1, \alpha_2, \dots, \alpha_{|W|}) \in R^{|W|},$$

where $\alpha_i = \sum_{j=0}^h (\lambda^j \cdot tfidf(w_i, u_{(t-j)}))$, u_t is the utterance mentioned in a turn t , $tfidf(w_i, u_t)$ is the product of term frequency of a word w_i in u_t and inverse document frequency of w_i , λ is a decay factor for giving more importance to more recent turns, $|W|$ is the size of word dictionary, and h is the number of previous turns considered as dialog history features.

After computing this relatedness between the current dialog context and every paragraph in the Wikipedia collection, two kernel structures are constructed using the information obtained from the highly-ranked paragraphs in the Wikipedia.

3.1 History Sequence Kernel

The first structure to be constructed for our composite kernel is a sequence of the most similar paragraph IDs of each turn from the beginning of the session to the current turn. Formally, the sequence S at a given turn t is defined as:

$$S = (s_0, \dots, s_t),$$

where $s_j = \text{argmax}_i (\text{sim}(x_j, p_i))$.

Since our hypothesis is that the more similar the dialog histories of the two inputs are, the more similar aspects of topic transitions occur for them, we propose a sub-sequence kernel (Lodhi et al., 2002) to map the data into a new feature space defined based on the similarity of each pair of history sequences as follows:

$$K_s(S_1, S_2) = \sum_{u \in \mathcal{A}^n} \sum_{i: u=S_1[i]} \sum_{j: u=S_2[j]} \lambda^{l(i)+l(j)},$$

where \mathcal{A} is a finite set of paragraph IDs, S is a finite sequence of paragraph IDs, u is a subsequence of S , $S[j]$ is the subsequence with the i -th characters $\forall i \in j$, $l(i)$ is the length of the subsequence, and $\lambda \in (0, 1)$ is a decay factor.

3.2 Domain Context Tree Kernel

The other kernel incorporates more various types of domain knowledge obtained from Wikipedia into the feature space. In this method, each instance is encoded in a tree structure constructed following the rules in Figure 2. The root node of a tree has few children, each of which is a subtree rooted at each paragraph node in:

$$\mathcal{P}_t = \{p_i | \text{sim}(x_t, p_i) > \theta\},$$

where θ is a threshold value to select the relevant paragraphs. Each subtree consists of a set of features from a given paragraph in the Wikipedia collection in a hierarchical structure. Figure 3 shows an example of a constructed tree.

Since this constructed tree structure represents semantic, discourse, and structural information extracted from the similar Wikipedia paragraphs to each given instance, we can explore these more enriched features to build the topic tracking model using a subset tree kernel (Collins and Duffy, 2002) which computes the similarity between each pair of trees in the feature space as follows:

$$K_t(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2),$$

where N_T is the set of T 's nodes, $\Delta(n_1, n_2) = \sum_i I_i(n_1) \cdot I_i(n_2)$, and $I_i(n)$ is a function that is 1 iff the i -th tree fragment occurs with root at node n and 0 otherwise.

3.3 Kernel Composition

In this work, a composite kernel is defined by combining the individual kernels including history sequence and domain context tree kernels, as well as

```

<TREE>:= (ROOT <PAR>...<PAR>)
<PAR>:= (PAR_ID <PARENTS>
        <PREV_PAR><NEXT_PAR><LINKS>)
<PARENTS>:= ('PARENTS' <ART><SEC>)
<ART>:= (ART_ID <ART_NAME><CAT_LIST>)
<ART_NAME>:= ('ART_NAME' ART_NAME)
<CAT_LIST>:= ('CAT' <CAT>...<CAT>)
<CAT>:= (CAT_ID *)
<SEC>:= (SEC_ID <SEC_NAME><PARENT_SEC>
        <PREV_SEC><NEXT_SEC>)
<SEC_NAME>:= ('SEC_NAME' SEC_NAME)
<PARENT_SEC>:= ('PRN_SEC', PRN_SEC_ID)
<PREV_SEC>:= ('PREV_SEC', PREV_SEC_NAME)
<NEXT_SEC>:= ('NEXT_SEC', NEXT_SEC_NAME)
<PREV_PAR>:= ('PREV_PAR', PREV_PAR_ID)
<NEXT_PAR>:= ('NEXT_PAR', NEXT_PAR_ID)
<LINKS>:= ('LINKS' <LINK>...<LINK>)
<LINK>:= (LINK_NAME *)

```

Figure 2: Rules for constructing a domain context tree from Wikipedia: PAR, ART, SEC, and CAT are acronyms for paragraph, article, section, and category, respectively

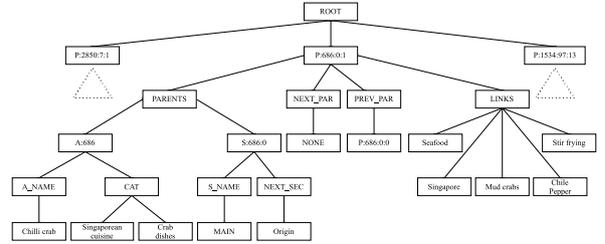


Figure 3: An example of domain context tree

the linear kernel between the vectors representing fundamental features extracted from the utterances themselves and the results of linguistic preprocessors. The composition is performed by linear combination as follows:

$$K(x_1, x_2) = \alpha \cdot K_l(V_1, V_2) + \beta \cdot K_s(S_1, S_2) + \gamma \cdot K_t(T_1, T_2),$$

where V_i , S_i , and T_i are the feature vector, history sequence, and domain context tree of x_i , respectively, K_l is the linear kernel computed by inner product of the vectors, α , β , and γ are coefficients for linear combination of three kernels, and $\alpha + \beta + \gamma = 1$.

4 Evaluation

To demonstrate the effectiveness of our proposed kernel method for dialog topic tracking, we performed experiments on the Singapore tour guide dialogs which consists of 35 dialog sessions collected from real human-human mixed initiative conversations related to Singapore between guides

and tourists. All the recorded dialogs with the total length of 21 hours were manually transcribed, then these transcribed dialogs with 19,651 utterances were manually annotated with the following nine topic categories: Opening, Closing, Itinerary, Accommodation, Attraction, Food, Transportation, Shopping, and Other.

Since we aim at developing the system which acts as a guide communicating with tourist users, an instance for both training and prediction of topic transition was created for each turn of tourists. The annotation of an instance is a pair of previous and current topics, and the actual number of labels occurred in the dataset is 65.

For each instance, the term vector was generated from the utterances in current user turn, previous system turn, and history turns within the window sizes $h = 10$. Then, the history sequence and tree context structures for our composite kernel were constructed based on 3,155 articles related to Singapore collected from Wikipedia database dump as of February 2013. For the linear kernel baseline, we used the following features: n-gram words, previous system actions, and current user acts which were manually annotated. Finally, 8,318 instances were used for training the model.

We trained the SVM models using SVM^{light}¹ (Joachims, 1999) with the following five different combinations of kernels: K_l only, K_l with \mathcal{P} as features, $K_l + K_s$, $K_l + K_t$, and $K_l + K_s + K_t$. The threshold value θ for selecting \mathcal{P} was 0.5, and the combinations of kernels were performed with the same α , β , or γ coefficient values for all sub-kernels. All the evaluations were done in five-fold cross validation to the manual annotations with two different metrics: one is accuracy of the predicted topic label for every turn, and the other is precision/recall/F-measure for each event of topic transition occurred either in the answer or the predicted result.

Table 1 compares the performances of the five combinations of kernels. When just the paragraph IDs were included as additional features, it failed to improve the performances from the baseline without any external features. However, our proposed kernels using history sequences and domain context trees achieved significant performances improvements for both evaluation metrics. While the history sequence kernel enhanced the coverage of the model to detect topic transitions,

¹<http://svmlight.joachims.org/>

	Turn-level	Transition-level		
	Accuracy	P	R	F
K_l	62.45	42.77	24.77	31.37
$K_l + \mathcal{P}$	62.44	42.76	24.77	31.37
$K_l + K_s$	67.19	39.94	40.59	40.26
$K_l + K_t$	68.54	45.55	35.69	40.02
All	69.98	44.82	39.83	42.18

Table 1: Experimental Results

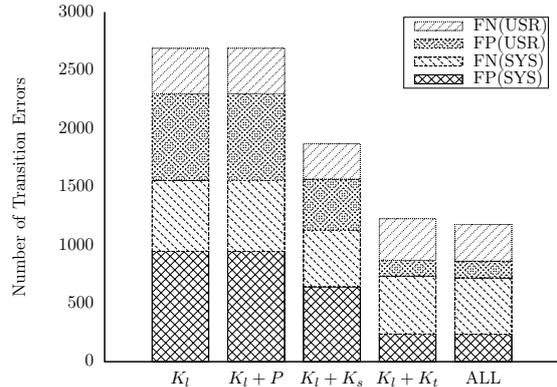


Figure 4: Error distributions of topic transitions: FN and FP denotes false negative and false positive respectively. USR and SYS in the parentheses indicate the initiativity of the transitions.

the domain context tree kernel contributed to produce more precise outputs. Finally, the model combining all the kernels outperformed the baseline by 7.53% in turn-level accuracy and 10.81% in transition-level F-measure.

The error distributions in Figure 4 indicate that these performance improvements were achieved by resolving the errors not only on user-initiative topic transitions, but also on system-initiative cases, which implies the effectiveness of the structured knowledge from Wikipedia to track the topics in mixed-initiative dialogs.

5 Conclusions

This paper presented a composite kernel approach for dialog topic tracking. This approach aimed to represent various types of domain knowledge obtained from Wikipedia as two structures: history sequences and domain context trees; then incorporate them into the model with kernel methods. Experimental results show that the proposed approaches helped to improve the topic tracking performances in mixed-initiative human-human dialogs with respect to the baseline model.

References

- P. H. Adams and C. H. Martell. 2008. Topic detection and extraction in chat. In *Proceedings of the 2008 IEEE International Conference on Semantic Computing*, pages 581–588.
- D. Bohus and A. Rudnicky. 2003. Ravenclaw: dialog management using hierarchical task decomposition and an expectation agenda. In *Proceedings of the European Conference on Speech, Communication and Technology*, pages 597–600.
- A. Breuing, U. Waltinger, and I. Wachsmuth. 2011. Harvesting wikipedia knowledge to identify topics in ongoing natural language dialogs. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pages 445–450.
- A. Celikyilmaz, D. Hakkani-Tür, and G. Tür. 2011. Approximate inference for domain detection in spoken language understanding. In *Proceedings of the 12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 713–716.
- Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 263–270.
- S. Ikeda, K. Komatani, T. Ogata, H. G. Okuno, and H. G. Okuno. 2008. Extensibility verification of robust domain selection against out-of-grammar utterances in multi-domain spoken dialogue system. In *Proceedings of the 9th INTERSPEECH*, pages 487–490.
- T. Joachims. 1999. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 11, pages 169–184. MIT Press, Cambridge, MA.
- K. Lagus and J. Kuusisto. 2002. Topic identification in natural language dialogues using neural networks. In *Proceedings of the 3rd SIGdial workshop on Discourse and dialogue*, pages 95–102.
- C. Lee, S. Jung, and G. G. Lee. 2008. Robust dialog management with n-best hypotheses using dialog examples and agenda. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 630–637.
- B. Lin, H. Wang, and L. Lee. 1999. A distributed architecture for cooperative spoken dialogue agents with coherent dialogue state and history. In *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*.
- Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. 2002. Text classification using string kernels. *The Journal of Machine Learning Research*, 2:419–444.
- T. Nakata, S. Ando, and A. Okumura. 2002. Topic detection based on dialogue history. In *Proceedings of the 19th international conference on Computational linguistics (COLING)*, pages 1–7.
- S. Roy and L. V. Subramaniam. 2006. Automatic generation of domain models for call centers from noisy transcriptions. In *Proceedings of COLING/ACL*, pages 737–744.
- G. Wilcock. 2012. Wikitalk: a spoken wikipedia-based open-domain knowledge access system. In *Proceedings of the Workshop on Question Answering for Complex Domains*, page 5770.
- S. Young, J. Schatzmann, K. Weilhammer, and H. Ye. 2007. The hidden information state approach to dialog management. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 149–152.
- Min Zhang, Jie Zhang, Jian Su, and Guodong Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 825–832.
- Shubin Zhao and Ralph Grishman. 2005. Extracting relations with integrated information using kernel methods. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 419–426.

An Extension of BLANC to System Mentions

Xiaoqiang Luo

Google Inc.

111 8th Ave, New York, NY 10011

xql@google.com

Sameer Pradhan

Harvard Medical School

300 Longwood Ave., Boston, MA 02115

sameer.pradhan@childrens.harvard.edu

Marta Recasens

Google Inc.

1600 Amphitheatre Pkwy,

Mountain View, CA 94043

recasens@google.com

Eduard Hovy

Carnegie Mellon University

5000 Forbes Ave.

Pittsburgh, PA 15213

hovy@cmu.edu

Abstract

BLANC is a link-based coreference evaluation metric for measuring the quality of coreference systems on gold mentions. This paper extends the original BLANC (“BLANC-gold” henceforth) to system mentions, removing the gold mention assumption. The proposed BLANC falls back seamlessly to the original one if system mentions are identical to gold mentions, and it is shown to strongly correlate with existing metrics on the 2011 and 2012 CoNLL data.

1 Introduction

Coreference resolution aims at identifying natural language expressions (or mentions) that refer to the same entity. It entails partitioning (often imperfect) mentions into equivalence classes. A critically important problem is how to measure the quality of a coreference resolution system. Many evaluation metrics have been proposed in the past two decades, including the MUC measure (Vilain et al., 1995), B-cubed (Bagga and Baldwin, 1998), CEAF (Luo, 2005) and, more recently, BLANC-gold (Recasens and Hovy, 2011). B-cubed and CEAF treat entities as sets of mentions and measure the agreement between key (or gold standard) entities and response (or system-generated) entities, while MUC and BLANC-gold are link-based.

In particular, MUC measures the degree of agreement between key coreference links (i.e., links among mentions within entities) and response coreference links, while non-coreference links (i.e., links formed by mentions from different entities) are not explicitly taken into account. This leads to a phenomenon where coreference systems outputting large entities are scored more favorably

than those outputting small entities (Luo, 2005). BLANC (Recasens and Hovy, 2011), on the other hand, considers both coreference links and non-coreference links. It calculates recall, precision and F-measure separately on coreference and non-coreference links in the usual way, and defines the overall recall, precision and F-measure as the mean of the respective measures for coreference and non-coreference links.

The BLANC-gold metric was developed with the assumption that response mentions and key mentions are identical. In reality, however, mentions need to be detected from natural language text and the result is, more often than not, imperfect: some key mentions may be missing in the response, and some response mentions may be spurious—so-called “twinless” mentions by Stoyanov et al. (2009). Therefore, the identical-mention-set assumption limits BLANC-gold’s applicability when gold mentions are not available, or when one wants to have a single score measuring both the quality of mention detection and coreference resolution. The goal of this paper is to extend the BLANC-gold metric to imperfect response mentions.

We first briefly review the original definition of BLANC, and rewrite its definition using set notation. We then argue that the gold-mention assumption in Recasens and Hovy (2011) can be lifted without changing the original definition. In fact, the proposed BLANC metric subsumes the original one in that its value is identical to the original one when response mentions are identical to key mentions.

The rest of the paper is organized as follows. We introduce the notions used in this paper in Section 2. We then present the original BLANC-gold in Section 3 using the set notation defined in Section 2. This paves the way to generalize it to

imperfect system mentions, which is presented in Section 4. The proposed BLANC is applied to the CoNLL 2011 and 2012 shared task participants, and the scores and its correlations with existing metrics are shown in Section 5.

2 Notations

To facilitate the presentation, we define the notations used in the paper.

We use *key* to refer to gold standard mentions or entities, and *response* to refer to system mentions or entities. The collection of *key* entities is denoted by $K = \{k_i\}_{i=1}^{|K|}$, where k_i is the i^{th} key entity; accordingly, $R = \{r_j\}_{j=1}^{|R|}$ is the set of *response* entities, and r_j is the j^{th} response entity. We assume that mentions in $\{k_i\}$ and $\{r_j\}$ are unique; in other words, there is no duplicate mention.

Let $C_k(i)$ and $C_r(j)$ be the set of *coreference* links formed by mentions in k_i and r_j :

$$\begin{aligned} C_k(i) &= \{(m_1, m_2) : m_1 \in k_i, m_2 \in k_i, m_1 \neq m_2\} \\ C_r(j) &= \{(m_1, m_2) : m_1 \in r_j, m_2 \in r_j, m_1 \neq m_2\} \end{aligned}$$

As can be seen, a link is an undirected edge between two mentions, and it can be equivalently represented by a pair of mentions. Note that when an entity consists of a single mention, its coreference link set is empty.

Let $N_k(i, j)$ ($i \neq j$) be *key non-coreference* links formed between mentions in k_i and those in k_j , and let $N_r(i, j)$ ($i \neq j$) be *response non-coreference* links formed between mentions in r_i and those in r_j , respectively:

$$\begin{aligned} N_k(i, j) &= \{(m_1, m_2) : m_1 \in k_i, m_2 \in k_j\} \\ N_r(i, j) &= \{(m_1, m_2) : m_1 \in r_i, m_2 \in r_j\} \end{aligned}$$

Note that the non-coreference link set is empty when all mentions are in the same entity.

We use the same letter and subscription without the index in parentheses to denote the union of sets, e.g.,

$$\begin{aligned} C_k &= \cup_i C_k(i), N_k = \cup_{i \neq j} N_k(i, j) \\ C_r &= \cup_j C_r(j), N_r = \cup_{i \neq j} N_r(i, j) \end{aligned}$$

We use $T_k = C_k \cup N_k$ and $T_r = C_r \cup N_r$ to denote the total set of key links and total set of response links, respectively. Clearly, C_k and N_k form a partition of T_k since $C_k \cap N_k = \emptyset$, $T_k = C_k \cup N_k$. Likewise, C_r and N_r form a partition of T_r .

We say that a key link $l_1 \in T_k$ equals a response link $l_2 \in T_r$ if and only if the pair of mentions from which the links are formed are identical. We write $l_1 = l_2$ if two links are equal. It is easy to see that the gold mention assumption—same set of response mentions as the set of key mentions—can be equivalently stated as $T_k = T_r$ (this does not necessarily mean that $C_k = C_r$ or $N_k = N_r$).

We also use $|\cdot|$ to denote the size of a set.

3 Original BLANC

BLANC-gold is adapted from Rand Index (Rand, 1971), a metric for clustering objects. Rand Index is defined as the ratio between the number of correct within-cluster links plus the number of correct cross-cluster links, and the total number of links.

When $T_k = T_r$, Rand Index can be applied directly since coreference resolution reduces to a clustering problem where mentions are partitioned into clusters (entities):

$$\text{Rand Index} = \frac{|C_k \cap C_r| + |N_k \cap N_r|}{\frac{1}{2}(|T_k|(|T_k| - 1))} \quad (1)$$

In practice, though, the simple-minded adoption of Rand Index is not satisfactory since the number of non-coreference links often overwhelms that of coreference links (Recasens and Hovy, 2011), or, $|N_k| \gg |C_k|$ and $|N_r| \gg |C_r|$. Rand Index, if used without modification, would not be sensitive to changes of coreference links.

BLANC-gold solves this problem by averaging the F-measure computed over coreference links and the F-measure over non-coreference links. Using the notations in Section 2, the recall, precision, and F-measure on coreference links are:

$$R_c^{(g)} = \frac{|C_k \cap C_r|}{|C_k \cap C_r| + |C_k \cap N_r|} \quad (2)$$

$$P_c^{(g)} = \frac{|C_k \cap C_r|}{|C_r \cap C_k| + |C_r \cap N_k|} \quad (3)$$

$$F_c^{(g)} = \frac{2R_c^{(g)}P_c^{(g)}}{R_c^{(g)} + P_c^{(g)}}; \quad (4)$$

Similarly, the recall, precision, and F-measure on non-coreference links are computed as:

$$R_n^{(g)} = \frac{|N_k \cap N_r|}{|N_k \cap C_r| + |N_k \cap N_r|} \quad (5)$$

$$P_n^{(g)} = \frac{|N_k \cap N_r|}{|N_r \cap C_k| + |N_r \cap N_k|} \quad (6)$$

$$F_n^{(g)} = \frac{2R_n^{(g)}P_n^{(g)}}{R_n^{(g)} + P_n^{(g)}}. \quad (7)$$

Finally, the BLANC-gold metric is the arithmetic average of $F_c^{(g)}$ and $F_n^{(g)}$:

$$\text{BLANC}^{(g)} = \frac{F_c^{(g)} + F_n^{(g)}}{2}. \quad (8)$$

Superscript g in these equations highlights the fact that they are meant for coreference systems with gold mentions.

Eqn. (8) indicates that BLANC-gold assigns equal weight to $F_c^{(g)}$, the F-measure from coreference links, and $F_n^{(g)}$, the F-measure from non-coreference links. This avoids the problem that $|N_k| \gg |C_k|$ and $|N_r| \gg |C_r|$, should the original Rand Index be used.

In Eqn. (2) - (3) and Eqn. (5) - (6), denominators are written as a sum of disjoint subsets so they can be related to the contingency table in (Recasens and Hovy, 2011). Under the assumption that $T_k = T_r$, it is clear that $C_k = (C_k \cap C_r) \cup (C_k \cap N_r)$, $C_r = (C_k \cap C_r) \cup (N_k \cap C_r)$, and so on.

4 BLANC for Imperfect Response Mentions

Under the assumption that the key and response mention sets are identical (which implies that $T_k = T_r$), Equations (2) to (7) make sense. For example, R_c is the ratio of the number of correct coreference links over the number of key coreference links; P_c is the ratio of the number of correct coreference links over the number of response coreference links, and so on.

However, when response mentions are not identical to key mentions, a key coreference link may not appear in either C_r or N_r , so Equations (2) to (7) cannot be applied directly to systems with imperfect mentions. For instance, if the key entities are $\{a, b, c\}$ $\{d, e\}$; and the response entities are $\{b, c\}$ $\{e, f, g\}$, then the key coreference link (a, b) is not seen on the response side; similarly, it is possible that a response link does not appear on the key side either: (c, f) and (f, g) are not in the key in the above example.

To account for missing or spurious links, we observe that

- $C_k \setminus T_r$ are key coreference links missing in the response;
- $N_k \setminus T_r$ are key non-coreference links missing in the response;
- $C_r \setminus T_k$ are response coreference links missing in the key;
- $N_r \setminus T_k$ are response non-coreference links

missing in the key,

and we propose to extend the coreference F-measure and non-coreference F-measure as follows. Coreference recall, precision and F-measure are changed to:

$$R_c = \frac{|C_k \cap C_r|}{|C_k \cap C_r| + |C_k \cap N_r| + |C_k \setminus T_r|} \quad (9)$$

$$P_c = \frac{|C_k \cap C_r|}{|C_r \cap C_k| + |C_r \cap N_k| + |C_r \setminus T_k|} \quad (10)$$

$$F_c = \frac{2R_c P_c}{R_c + P_c} \quad (11)$$

Non-coreference recall, precision and F-measure are changed to:

$$R_n = \frac{|N_k \cap N_r|}{|N_k \cap C_r| + |N_k \cap N_r| + |N_k \setminus T_r|} \quad (12)$$

$$P_n = \frac{|N_k \cap N_r|}{|N_r \cap C_k| + |N_r \cap N_k| + |N_r \setminus T_k|} \quad (13)$$

$$F_n = \frac{2R_n P_n}{R_n + P_n}. \quad (14)$$

The proposed BLANC continues to be the arithmetic average of F_c and F_n :

$$\text{BLANC} = \frac{F_c + F_n}{2}. \quad (15)$$

We observe that the definition of the proposed BLANC, Equ. (9)-(14) subsume the BLANC-gold (2) to (7) due to the following proposition: If $T_k = T_r$, then $\text{BLANC} = \text{BLANC}^{(g)}$.

Proof. We only need to show that $R_c = R_c^{(g)}$, $P_c = P_c^{(g)}$, $R_n = R_n^{(g)}$, and $P_n = P_n^{(g)}$. We prove the first one (the other proofs are similar and elided due to space limitations). Since $T_k = T_r$ and $C_k \subset T_k$, we have $C_k \subset T_r$; thus $C_k \setminus T_r = \emptyset$, and $|C_k \cap T_r| = 0$. This establishes that $R_c = R_c^{(g)}$.

Indeed, since C_k is a union of three disjoint subsets: $C_k = (C_k \cap C_r) \cup (C_k \cap N_r) \cup (C_k \setminus T_r)$, $R_c^{(g)}$ and R_c can be unified as $\frac{|C_k \cap C_r|}{|C_k|}$. Unification for other component recalls and precisions can be done similarly. So the final definition of BLANC can be succinctly stated as:

$$R_c = \frac{|C_k \cap C_r|}{|C_k|}, \quad P_c = \frac{|C_k \cap C_r|}{|C_r|} \quad (16)$$

$$R_n = \frac{|N_k \cap N_r|}{|N_k|}, \quad P_n = \frac{|N_k \cap N_r|}{|N_r|} \quad (17)$$

$$F_c = \frac{2|C_k \cap C_r|}{|C_k| + |C_r|}, \quad F_n = \frac{2|N_k \cap N_r|}{|N_k| + |N_r|} \quad (18)$$

$$\text{BLANC} = \frac{F_c + F_n}{2} \quad (19)$$

4.1 Boundary Cases

Care has to be taken when counts of the BLANC definition are 0. This can happen when all key (or response) mentions are in one cluster or are all singletons: the former case will lead to $N_k = \emptyset$ (or $N_r = \emptyset$); the latter will lead to $C_k = \emptyset$ (or $C_r = \emptyset$). Observe that as long as $|C_k| + |C_r| > 0$, F_c in (18) is well-defined; as long as $|N_k| + |N_r| > 0$, F_n in (18) is well-defined. So we only need to augment the BLANC definition for the following cases:

(1) If $C_k = C_r = \emptyset$ and $N_k = N_r = \emptyset$, then $\text{BLANC} = I(M_k = M_r)$, where $I(\cdot)$ is an indicator function whose value is 1 if its argument is true, and 0 otherwise. M_k and M_r are the key and response mention set. This can happen when a document has no more than one mention and there is no link.

(2) If $C_k = C_r = \emptyset$ and $|N_k| + |N_r| > 0$, then $\text{BLANC} = F_n$. This is the case where the key and response side has only entities consisting of singleton mentions. Since there is no coreference link, BLANC reduces to the non-coreference F-measure F_n .

(3) If $N_k = N_r = \emptyset$ and $|C_k| + |C_r| > 0$, then $\text{BLANC} = F_c$. This is the case where all mentions in the key and response are in one entity. Since there is no non-coreference link, BLANC reduces to the coreference F-measure F_c .

4.2 Toy Examples

We walk through a few examples and show how BLANC is calculated in detail. In all the examples below, each lower-case letter represents a mention; mentions in an entity are closed in $\{\}$; two letters in $()$ represent a link.

Example 1. Key entities are $\{abc\}$ and $\{d\}$; response entities are $\{bc\}$ and $\{de\}$. Obviously,

$$C_k = \{(ab), (bc), (ac)\};$$

$$N_k = \{(ad), (bd), (cd)\};$$

$$C_r = \{(bc), (de)\};$$

$$N_r = \{(bd), (be), (cd), (ce)\}.$$

Therefore, $C_k \cap C_r = \{(bc)\}$, $N_k \cap N_r = \{(bd), (cd)\}$, and $R_c = \frac{1}{3}$, $P_c = \frac{1}{2}$, $F_c = \frac{2}{5}$; $R_n = \frac{2}{3}$, $P_n = \frac{2}{4}$, $F_n = \frac{4}{7}$. Finally, $\text{BLANC} = \frac{17}{35}$.

Example 2. Key entity is $\{a\}$; response entity is $\{b\}$. This is boundary case (1): $\text{BLANC} = 0$.

Example 3. Key entities are $\{a\}\{b\}\{c\}$; response entities are $\{a\}\{b\}\{d\}$. This is boundary case (2): there are no coreference links. Since

$$N_k = \{(ab), (bc), (ca)\},$$

Participant	R	P	BLANC
lee	50.23	49.28	48.84
sapena	40.68	49.05	44.47
nugues	47.83	44.22	45.95
chang	44.71	47.48	45.49
stoyanov	49.37	29.80	34.58
santos	46.74	37.33	41.33
song	36.88	39.69	30.92
sobha	35.42	39.56	36.31
yang	47.95	29.12	36.09
charton	42.32	31.54	35.65
hao	45.41	32.75	36.98
zhou	29.93	45.58	34.95
kobdani	32.29	33.01	32.57
xinxin	36.83	34.39	35.02
kummerfeld	34.84	29.53	30.98
zhang	30.10	43.96	35.71
zhakova	26.40	15.32	15.37
irwin	3.62	28.28	6.28

Table 1: The proposed BLANC scores of the CoNLL-2011 shared task participants.

$$N_r = \{(ab), (bd), (ad)\},$$

we have

$$N_k \cap N_r = \{(ab)\}, \text{ and } R_n = \frac{1}{3}, P_n = \frac{1}{3}.$$

So $\text{BLANC} = F_n = \frac{1}{3}$.

Example 4. Key entity is $\{abc\}$; response entity is $\{bc\}$. This is boundary case (3): there are no non-coreference links. Since

$$C_k = \{(ab), (bc), (ca)\}, \text{ and } C_r = \{(bc)\},$$

we have

$$C_k \cap C_r = \{(bc)\}, \text{ and } R_c = \frac{1}{3}, P_c = 1,$$

So $\text{BLANC} = F_c = \frac{2}{4} = \frac{1}{2}$.

5 Results

5.1 CoNLL-2011/12

We have updated the publicly available CoNLL coreference scorer¹ with the proposed BLANC, and used it to compute the proposed BLANC scores for all the CoNLL 2011 (Pradhan et al., 2011) and 2012 (Pradhan et al., 2012) participants in the official track, where participants had to automatically predict the mentions. Tables 1 and 2 report the updated results.²

5.2 Correlation with Other Measures

Figure 1 shows how the proposed BLANC measure works when compared with existing metrics such as MUC, B-cubed and CEAF, using the BLANC and F1 scores. The proposed BLANC is highly positively correlated with the

¹<http://code.google.com/p/reference-coreference-scorers>

²The order is kept the same as in Pradhan et al. (2011) and Pradhan et al. (2012) for easy comparison.

Participant	R	P	BLANC
Language: Arabic			
fernandes	33.43	44.66	37.99
bjorkelund	32.65	45.47	37.93
uryupina	31.62	35.26	33.02
stamborg	32.59	36.92	34.50
chen	31.81	31.52	30.82
zhekova	11.04	62.58	18.51
li	4.60	56.63	8.42
Language: English			
fernandes	54.91	63.66	58.75
martschat	52.00	58.84	55.04
bjorkelund	52.01	59.55	55.42
chang	52.85	55.03	53.86
chen	50.52	56.82	52.87
chunyang	51.19	55.47	52.65
stamborg	54.39	54.88	54.42
yuan	50.58	54.29	52.11
xu	45.99	54.59	46.47
shou	49.55	52.46	50.44
uryupina	44.15	48.89	46.04
songyang	40.60	50.85	45.10
zhekova	41.46	33.13	34.80
xinxin	44.39	32.79	36.54
li	25.17	52.96	31.85
Language: Chinese			
chen	48.45	62.44	54.10
yuan	53.15	40.75	43.20
bjorkelund	47.58	45.93	44.22
xu	44.11	36.45	38.45
fernandes	42.36	61.72	49.63
stamborg	39.60	55.12	45.89
uryupina	33.44	56.01	41.88
martschat	27.24	62.33	37.89
chunyang	37.43	36.18	36.77
xinxin	36.46	39.79	37.85
li	21.61	62.94	30.37
chang	18.74	40.76	25.68
zhekova	21.50	37.18	22.89

Table 2: The proposed BLANC scores of the CoNLL-2012 shared task participants.

	R	P	F1
MUC	0.975	0.844	0.935
B-cubed	0.981	0.942	0.966
CEAF-m	0.941	0.923	0.966
CEAF-e	0.797	0.781	0.919

Table 3: Pearson’s r correlation coefficients between the proposed BLANC and the other coreference measures based on the CoNLL 2011/2012 results. All p -values are significant at < 0.001 .

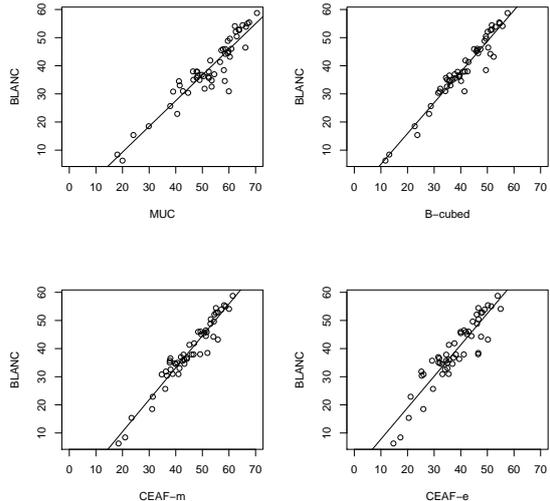


Figure 1: Correlation plot between the proposed BLANC and the other measures based on the CoNLL 2011/2012 results. All values are F1 scores.

other measures along R, P and F1 (Table 3), showing that BLANC is able to capture most entity-based similarities measured by B-cubed and CEAF. However, the CoNLL data sets come from OntoNotes (Hovy et al., 2006), where singleton entities are not annotated, and BLANC has a wider dynamic range on data sets with singletons (Recasens and Hovy, 2011). So the correlations will likely be lower on data sets with singleton entities.

6 Conclusion

The original BLANC-gold (Recasens and Hovy, 2011) requires that system mentions be identical to gold mentions, which limits the metric’s utility since detected system mentions often have missing key mentions or spurious mentions. The proposed BLANC is free from this assumption, and we have shown that it subsumes the original BLANC-gold. Since BLANC works on imperfect system mentions, we have used it to score the CoNLL 2011 and 2012 coreference systems. The BLANC scores show strong correlation with existing metrics, especially B-cubed and CEAF-m.

Acknowledgments

We would like to thank the three anonymous reviewers for their invaluable suggestions for improving the paper. This work was partially supported by grants R01LM10090 from the National Library of Medicine.

References

- Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the Linguistic Coreference Workshop at The First International Conference on Language Resources and Evaluation (LREC'98)*, pages 563–566.
- Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, USA, June. Association for Computational Linguistics.
- Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proc. of Human Language Technology (HLT)/Empirical Methods in Natural Language Processing (EMNLP)*.
- Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. 2011. CoNLL-2011 shared task: Modeling unrestricted coreference in OntoNotes. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–27, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea, July. Association for Computational Linguistics.
- W. M. Rand. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.
- M. Recasens and E. Hovy. 2011. BLANC: Implementing the Rand index for coreference evaluation. *Natural Language Engineering*, 17:485–510, 10.
- Veselin Stoyanov, Nathan Gilbert, Claire Cardie, and Ellen Riloff. 2009. Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. 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: Volume 2 - Volume 2*, ACL '09, pages 656–664, Stroudsburg, PA, USA. Association for Computational Linguistics.
- M. Vilain, J. Burger, J. Aberdeen, D. Connolly, , and L. Hirschman. 1995. A model-theoretic coreference scoring scheme. In *In Proc. of MUC6*, pages 45–52.

Scoring Coreference Partitions of Predicted Mentions: A Reference Implementation

Sameer Pradhan¹, Xiaoqiang Luo², Marta Recasens³,
Eduard Hovy⁴, Vincent Ng⁵ and Michael Strube⁶

¹Harvard Medical School, Boston, MA, ²Google Inc., New York, NY

³Google Inc., Mountain View, CA, ⁴Carnegie Mellon University, Pittsburgh, PA

⁵HLTRI, University of Texas at Dallas, Richardson, TX, ⁶HITS, Heidelberg, Germany

sameer.pradhan@childrens.harvard.edu, {xql, recasens}@google.com,
hovy@cmu.edu, vince@hlt.utdallas.edu, michael.strube@h-its.org

Abstract

The definitions of two coreference scoring metrics— B^3 and CEAf—are underspecified with respect to *predicted*, as opposed to *key* (or *gold*) mentions. Several variations have been proposed that manipulate either, or both, the key and predicted mentions in order to get a one-to-one mapping. On the other hand, the metric BLANC was, until recently, limited to scoring partitions of key mentions. In this paper, we (i) argue that mention manipulation for scoring predicted mentions is unnecessary, and potentially harmful as it could produce unintuitive results; (ii) illustrate the application of all these measures to scoring predicted mentions; (iii) make available an open-source, thoroughly-tested reference implementation of the main coreference evaluation measures; and (iv) rescore the results of the CoNLL-2011/2012 shared task systems with this implementation. This will help the community accurately measure and compare new end-to-end coreference resolution algorithms.

1 Introduction

Coreference resolution is a key task in natural language processing (Jurafsky and Martin, 2008) aiming to detect the referential expressions (*mentions*) in a text that point to the same entity. Roughly over the past two decades, research in coreference (for the English language) had been plagued by individually crafted evaluations based on two central corpora—MUC (Hirschman and Chinchor, 1997; Chinchor and Sundheim, 2003; Chinchor, 2001) and ACE (Doddington et al., 2004). Experimental parameters ranged from using perfect (*gold*, or *key*) mentions as input for

purely testing the quality of the entity linking algorithm, to an end-to-end evaluation where *predicted mentions* are used. Given the range of evaluation parameters and disparity between the annotation standards for the two corpora, it was very hard to grasp the state of the art for the task of coreference. This has been expounded in Stoyanov et al. (2009). The activity in this sub-field of NLP can be gauged by: (i) the continual addition of corpora manually annotated for coreference—The OntoNotes corpus (Pradhan et al., 2007; Weischedel et al., 2011) in the general domain, as well as the i2b2 (Uzuner et al., 2012) and THYME (Styler et al., 2014) corpora in the clinical domain would be a few examples of such emerging corpora; and (ii) ongoing proposals for refining the existing metrics to make them more informative (Holen, 2013; Chen and Ng, 2013).

The CoNLL-2011/2012 shared tasks on coreference resolution using the OntoNotes corpus (Pradhan et al., 2011; Pradhan et al., 2012) were an attempt to standardize the evaluation settings by providing a benchmark annotated corpus, scorer, and state-of-the-art system results that would allow future systems to compare against them. Following the timely emphasis on end-to-end evaluation, the official track used predicted mentions and measured performance using five coreference measures: MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), $CEAF_e$ (Luo, 2005), $CEAF_m$ (Luo, 2005), and BLANC (Recasens and Hovy, 2011). The arithmetic mean of the first three was the task’s final score.

An unfortunate setback to these evaluations had its root in three issues: (i) the multiple variations of two of the scoring metrics— B^3 and CEAf—used by the community to handle predicted mentions; (ii) a buggy implementation of the Cai and Strube (2010) proposal that tried to reconcile these variations; and (iii) the erroneous computation of

the BLANC metric for partitions of predicted mentions. Different interpretations as to how to compute B^3 and CEAFF scores for coreference systems when predicted mentions do not perfectly align with key mentions—which is usually the case—led to variations of these metrics that manipulate the gold standard and system output in order to get a one-to-one mention mapping (Stoyanov et al., 2009; Cai and Strube, 2010). Some of these variations arguably produce rather unintuitive results, while others are not faithful to the original measures.

In this paper, we address the issues in scoring coreference partitions of predicted mentions. Specifically, we justify our decision to go back to the original scoring algorithms by arguing that manipulation of key or predicted mentions is unnecessary and could in fact produce unintuitive results. We demonstrate the use of our recent extension of BLANC that can seamlessly handle predicted mentions (Luo et al., 2014). We make available an open-source, thoroughly-tested reference implementation of the main coreference evaluation measures that do not involve mention manipulation and is faithful to the original intentions of the proposers of these metrics. We republish the CoNLL-2011/2012 results based on this scorer, so that future systems can use it for evaluation and have the CoNLL results available for comparison.

The rest of the paper is organized as follows. Section 2 provides an overview of the variations of the existing measures. We present our newly updated coreference scoring package in Section 3 together with the rescored CoNLL-2011/2012 outputs. Section 4 walks through a scoring example for all the measures, and we conclude in Section 5.

2 Variations of Scoring Measures

Two commonly used coreference scoring metrics— B^3 and CEAFF—are underspecified in their application for scoring *predicted*, as opposed to *key* mentions. The examples in the papers describing these metrics assume perfect mentions where predicted mentions are the same set of mentions as key mentions. The lack of accompanying reference implementation for these metrics by its proposers made it harder to fill the gaps in the specification. Subsequently, different interpretations of how one can evaluate coreference systems when predicted mentions do not perfectly align with key mentions led to variations of these metrics that manipulate the gold and/or predicted mentions (Stoy-

anov et al., 2009; Cai and Strube, 2010). All these variations attempted to generate a one-to-one mapping between the key and predicted mentions, assuming that the original measures cannot be applied to predicted mentions. Below we first provide an overview of these variations and then discuss the unnecessary of this assumption.

Coining the term *twinless mentions* for those mentions that are either spurious or missing from the predicted mention set, Stoyanov et al. (2009) proposed two variations to B^3 — B_{all}^3 and B_0^3 —to handle them. In the first variation, all predicted twinless mentions are retained, whereas the latter discards them and penalizes recall for twinless predicted mentions. Rahman and Ng (2009) proposed another variation by removing “all and only those twinless system mentions that are singletons before applying B^3 and CEAFF.” Following upon this line of research, Cai and Strube (2010) proposed a unified solution for both B^3 and $CEAF_m$, leaving the question of handling $CEAF_e$ as future work because “it produces unintuitive results.” The essence of their solution involves manipulating twinless key and predicted mentions by adding them either from the predicted partition to the key partition or vice versa, depending on whether one is computing precision or recall. The Cai and Strube (2010) variation was used by the CoNLL-2011/2012 shared tasks on coreference resolution using the OntoNotes corpus, and by the i2b2 2011 shared task on coreference resolution using an assortment of clinical notes corpora (Uzuner et al., 2012).¹ It was later identified by Recasens et al. (2013) that there was a bug in the implementation of this variation in the scorer used for the CoNLL-2011/2012 tasks. We have not tested the correctness of this variation in the scoring package used for the i2b2 shared task.

However, it turns out that the CEAFF metric (Luo, 2005) was always intended to work seamlessly on predicted mentions, and so has been the case with the B^3 metric.² In a latter paper, Rahman and Ng (2011) correctly state that “CEAFF can compare partitions with twinless mentions without any modification.” We will look at this further in Section 4.3.

We argue that manipulations of key and response mentions/entities, as is done in the existing B^3 variations, not only confound the evaluation process, but are also subject to abuse and can seriously jeopardize the fidelity of the evalu-

¹Personal communication with Andreea Bodnari, and contents of the i2b2 scorer code.

²Personal communication with Breck Baldwin.

ation. Given space constraints we use an example worked out in Cai and Strube (2010). Let the key contain an entity with mentions $\{a, b, c\}$ and the prediction contain an entity with mentions $\{a, b, d\}$. As detailed in Cai and Strube (2010, p. 29-30, Tables 1–3), B_0^3 assigns a perfect precision of 1.00 which is unintuitive as the system has wrongly predicted a mention d as belonging to the entity. For the same prediction, B_{all}^3 assigns a precision of 0.556. But, if the prediction contains two entities $\{a, b, d\}$ and $\{c\}$ (i.e., the mention c is added as a spurious singleton), then B_{all}^3 precision increases to 0.667 which is counter-intuitive as it does not penalize the fact that c is erroneously placed in its own entity. The version illustrated in Section 4.2, which is devoid of any mention manipulations, gives a precision of 0.444 in the first scenario and the precision drops to 0.333 in the second scenario with the addition of a spurious singleton entity $\{c\}$. This is a more intuitive behavior.

Contrary to both B^3 and CEF, the BLANC measure (Recasens and Hovy, 2011) was never designed to handle predicted mentions. However, the implementation used for the SemEval-2010 shared task as well as the one for the CoNLL-2011/2012 shared tasks accepted predicted mentions as input, producing undefined results. In Luo et al. (2014) we have extended the BLANC metric to deal with predicted mentions

3 Reference Implementation

Given the potential unintuitive outcomes of mention manipulation and the misunderstanding that the original measures could not handle twinless predicted mentions (Section 2), we redesigned the CoNLL scorer. The new implementation:

- is faithful to the original measures;
- removes any prior mention manipulation, which might depend on specific annotation guidelines among other problems;
- has been thoroughly tested to ensure that it gives the expected results according to the original papers, and all test cases are included as part of the release;
- is free of the reported bugs that the CoNLL scorer (v4) suffered (Recasens et al., 2013);
- includes the extension of BLANC to handle predicted mentions (Luo et al., 2014).

This is the open source scoring package³ that we present as a reference implementation for the

³<http://code.google.com/p/reference-coreference-scorers/>

SYSTEM	MD	MUC	B^3	CEAF		BLANC	CONLL AVERAGE
				m	e		
	F_1	F_1^1	F_1^2	F_1	F_1^3		
CoNLL-2011; English							
lee	70.7	59.6	48.9	53.0	46.1	48.8	51.5
sapena	68.4	59.5	46.5	51.3	44.0	44.5	50.0
nugues	69.0	58.6	45.0	48.4	40.0	46.0	47.9
chang	64.9	57.2	46.0	50.7	40.0	45.5	47.7
stoyanov	67.8	58.4	40.1	43.3	36.9	34.6	45.1
santos	65.5	56.7	42.9	45.1	35.6	41.3	45.0
song	67.3	60.0	41.4	41.0	33.1	30.9	44.8
sobha	64.8	50.5	39.5	44.2	39.4	36.3	43.1
yang	63.9	52.3	39.4	43.2	35.5	36.1	42.4
charton	64.3	52.5	38.0	42.6	34.5	35.7	41.6
hao	64.3	54.5	37.7	41.9	31.6	37.0	41.3
zhou	62.3	49.0	37.0	40.6	35.0	35.0	40.3
kobdani	61.0	53.5	34.8	38.1	34.1	32.6	38.7
xinxin	61.9	46.6	34.9	37.7	31.7	35.0	37.7
kummerfeld	62.7	42.7	34.2	38.8	35.5	31.0	37.5
zhang	61.1	47.9	34.4	37.8	29.2	35.7	37.2
zhekova	48.3	24.1	23.7	23.4	20.5	15.4	22.8
irwin	26.7	20.0	11.7	18.5	14.7	6.3	15.5
CoNLL-2012; English							
fernandes	77.7	70.5	57.6	61.4	53.9	58.8	60.7
martschat	75.2	67.0	54.6	58.8	51.5	55.0	57.7
bjorkelund	75.4	67.6	54.5	58.2	50.2	55.4	57.4
chang	74.3	66.4	53.0	57.1	48.9	53.9	56.1
chen	73.8	63.7	51.8	55.8	48.1	52.9	54.5
chunyang	73.7	63.8	51.2	55.1	47.6	52.7	54.2
stamborg	73.9	65.1	51.7	55.1	46.6	54.4	54.2
yuan	72.5	62.6	50.1	54.5	46.0	52.1	52.9
xu	72.0	66.2	50.3	51.3	41.3	46.5	52.6
shou	73.7	62.9	49.4	53.2	46.7	50.4	53.0
uryupina	70.9	60.9	46.2	49.3	42.9	46.0	50.0
songyang	68.8	59.8	45.9	49.6	42.4	45.1	49.4
zhekova	67.1	53.5	35.7	39.7	32.2	34.8	40.5
xinxin	62.8	48.3	35.7	38.0	31.9	36.5	38.6
li	59.9	50.8	32.3	36.3	25.2	31.9	36.1
CoNLL-2012; Chinese							
chen	71.6	62.2	55.7	60.0	55.0	54.1	57.6
yuan	68.2	60.3	52.4	55.8	50.2	43.2	54.3
bjorkelund	66.4	58.6	51.1	54.2	47.6	44.2	52.5
xu	65.2	58.1	49.5	51.9	46.6	38.5	51.4
fernandes	66.1	60.3	49.6	54.4	44.5	49.6	51.5
stamborg	64.0	57.8	47.4	51.6	41.9	45.9	49.0
uryupina	59.0	53.0	41.7	46.9	37.6	41.9	44.1
martschat	58.6	52.4	40.8	46.0	38.2	37.9	43.8
chunyang	61.6	49.8	39.6	44.2	37.3	36.8	42.2
xinxin	55.9	48.1	38.8	42.9	34.5	37.9	40.5
li	51.5	44.7	31.5	36.7	25.3	30.4	33.8
chang	47.6	37.9	28.8	36.1	29.6	25.7	32.1
zhekova	47.3	40.6	28.1	31.4	21.2	22.9	30.0
CoNLL-2012; Arabic							
fernandes	64.8	46.5	42.5	49.2	46.5	38.0	45.2
bjorkelund	60.6	47.8	41.6	46.7	41.2	37.9	43.5
uryupina	55.4	41.5	36.1	41.4	35.0	33.0	37.5
stamborg	59.5	41.2	35.9	40.0	32.9	34.5	36.7
chen	59.8	39.0	32.1	34.7	26.0	30.8	32.4
zhekova	41.0	29.9	22.7	31.1	25.9	18.5	26.2
li	29.7	18.1	13.1	21.0	17.3	8.4	16.2

Table 1: Performance on the **official, closed** track in percentages using all predicted information for the CoNLL-2011 and 2012 shared tasks.

community to use. It is written in perl and stems from the scorer that was initially used for the SemEval-2010 shared task (Recasens et al., 2010) and later modified for the CoNLL-2011/2012 shared tasks.⁴

Partitioning detected mentions into entities (or equivalence classes) typically comprises two distinct tasks: (i) mention detection; and (ii) coreference resolution. A typical two-step coreference algorithm uses mentions generated by the best

⁴We would like to thank Emili Sapena for writing the first version of the scoring package.

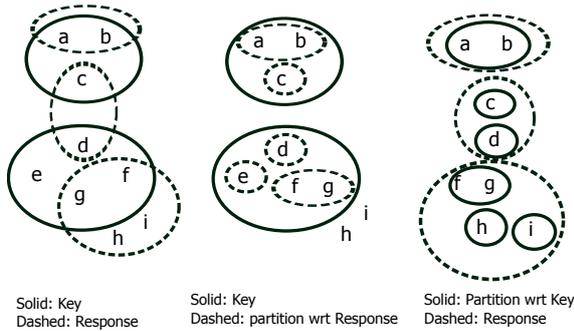


Figure 1: Example key and response entities along with the partitions for computing the MUC score.

possible mention detection algorithm as input to the coreference algorithm. Therefore, ideally one would want to score the two steps independently of each other. A peculiarity of the OntoNotes corpus is that singleton referential mentions are not annotated, thereby preventing the computation of a mention detection score independently of the coreference resolution score. In corpora where all referential mentions (including singletons) are annotated, the mention detection score generated by this implementation is independent of the coreference resolution score.

We used this reference implementation to rescore the CoNLL-2011/2012 system outputs for the official task to enable future comparisons with these benchmarks. The new CoNLL-2011/2012 results are in Table 1. We found that the overall system ranking remained largely unchanged for both shared tasks, except for some of the lower ranking systems that changed one or two places. However, there was a considerable drop in the magnitude of all B^3 scores owing to the combination of two things: (i) mention manipulation, as proposed by Cai and Strube (2010), adds singletons to account for *twinless* mentions; and (ii) the B^3 metric allows an entity to be used more than once as pointed out by Luo (2005). This resulted in a drop in the CoNLL averages (B^3 is one of the three measures that make the average).

4 An Illustrative Example

This section walks through the process of computing each of the commonly used metrics for an example where the set of predicted mentions has some missing key mentions and some spurious mentions. While the mathematical formulae for these metrics can be found in the original papers (Vilain et al., 1995; Bagga and Baldwin,

1998; Luo, 2005), many misunderstandings discussed in Section 2 are due to the fact that these papers lack an example showing how a metric is computed on predicted mentions. A concrete example goes a long way to prevent similar misunderstandings in the future. The example is adapted from Vilain et al. (1995) with some slight modifications so that the total number of mentions in the key is different from the number of mentions in the prediction. The key (K) contains two entities with mentions $\{a, b, c\}$ and $\{d, e, f, g\}$ and the response (R) contains three entities with mentions $\{a, b\}$; $\{c, d\}$ and $\{f, g, h, i\}$:

$$K = \overbrace{\{a, b, c\}}^{K_1} \overbrace{\{d, e, f, g\}}^{K_2} \quad (1)$$

$$R = \overbrace{\{a, b\}}^{R_1} \overbrace{\{c, d\}}^{R_2} \overbrace{\{f, g, h, i\}}^{R_3}. \quad (2)$$

Mention e is missing from the response, and mentions h and i are spurious in the response. The following sections use R to denote recall and P for precision.

4.1 MUC

The main step in the MUC scoring is creating the partitions with respect to the key and response respectively, as shown in Figure 1. Once we have the partitions, then we compute the MUC score by:

$$\begin{aligned}
 R &= \frac{\sum_{i=1}^{N_k} (|K_i| - |p(K_i)|)}{\sum_{i=1}^{N_k} (|K_i| - 1)} \\
 &= \frac{(3 - 2) + (4 - 3)}{(3 - 1) + (4 - 1)} = 0.40 \\
 P &= \frac{\sum_{i=1}^{N_r} (|R_i| - |p'(R_i)|)}{\sum_{i=1}^{N_r} (|R_i| - 1)} \\
 &= \frac{(2 - 1) + (2 - 2) + (4 - 3)}{(2 - 1) + (2 - 1) + (4 - 1)} = 0.40,
 \end{aligned}$$

where K_i is the i^{th} key entity and $p(K_i)$ is the set of partitions created by intersecting K_i with response entities (cf. the middle sub-figure in Figure 1); R_i is the i^{th} response entity and $p'(R_i)$ is the set of partitions created by intersecting R_i with key entities (cf. the right-most sub-figure in Figure 1); and N_k and N_r are the number of key and response entities, respectively.

The MUC F_1 score in this case is 0.40.

4.2 B^3

For computing B^3 recall, each key mention is assigned a credit equal to the ratio of the number of correct mentions in the predicted entity *containing* the key mention to the size of the key entity to which the mention belongs, and the recall is just

the sum of credits over all key mentions normalized over the number of key mentions. B³ precision is computed similarly, except switching the role of key and response. Applied to the example:

$$\begin{aligned}
R &= \frac{\sum_{i=1}^{N_k} \sum_{j=1}^{N_r} \frac{|K_i \cap R_j|^2}{|K_i|}}{\sum_{i=1}^{N_k} |K_i|} \\
&= \frac{1}{7} \times \left(\frac{2^2}{3} + \frac{1^2}{3} + \frac{1^2}{4} + \frac{2^2}{4} \right) = \frac{1}{7} \times \frac{35}{12} \approx 0.42 \\
P &= \frac{\sum_{i=1}^{N_k} \sum_{j=1}^{N_r} \frac{|K_i \cap R_j|^2}{|R_j|}}{\sum_{i=1}^{N_r} |R_j|} \\
&= \frac{1}{8} \times \left(\frac{2^2}{2} + \frac{1^2}{2} + \frac{1^2}{2} + \frac{2^2}{4} \right) = \frac{1}{8} \times \frac{4}{1} = 0.50
\end{aligned}$$

Note that terms with 0 value are omitted. The B³ F₁ score is 0.46.

4.3 CEAf

The first step in the CEAf computation is getting the best scoring alignment between the key and response entities. In this case the alignment is straightforward. Entity R_1 aligns with K_1 and R_3 aligns with K_2 . R_2 remains unaligned.

CEAF_m

CEAF_m recall is the number of aligned mentions divided by the number of key mentions, and precision is the number of aligned mentions divided by the number of response mentions:

$$\begin{aligned}
R &= \frac{|K_1 \cap R_1| + |K_2 \cap R_3|}{|K_1| + |K_2|} = \frac{(2+2)}{(3+4)} \approx 0.57 \\
P &= \frac{|K_1 \cap R_1| + |K_2 \cap R_3|}{|R_1| + |R_2| + |R_3|} = \frac{(2+2)}{(2+2+4)} = 0.50
\end{aligned}$$

The CEAf_m F₁ score is 0.53.

CEAF_e

We use the same notation as in Luo (2005): $\phi_4(K_i, R_j)$ to denote the similarity between a key entity K_i and a response entity R_j . $\phi_4(K_i, R_j)$ is defined as:

$$\phi_4(K_i, R_j) = \frac{2 \times |K_i \cap R_j|}{|K_i| + |R_j|}.$$

CEAF_e recall and precision, when applied to this example, are:

$$\begin{aligned}
R &= \frac{\phi_4(K_1, R_1) + \phi_4(K_2, R_3)}{N_k} = \frac{\frac{(2 \times 2)}{(3+2)} + \frac{(2 \times 2)}{(4+4)}}{2} = 0.65 \\
P &= \frac{\phi_4(K_1, R_1) + \phi_4(K_2, R_3)}{N_r} = \frac{\frac{(2 \times 2)}{(3+2)} + \frac{(2 \times 2)}{(4+4)}}{3} \approx 0.43
\end{aligned}$$

The CEAf_e F₁ score is 0.52.

4.4 BLANC

The BLANC metric illustrated here is the one in our implementation which extends the original

BLANC (Recasens and Hovy, 2011) to predicted mentions (Luo et al., 2014).

Let C_k and C_r be the set of coreference links in the key and response respectively, and N_k and N_r be the set of non-coreference links in the key and response respectively. A link between a mention pair m and n is denoted by mn ; then for the example in Figure 1, we have

$$\begin{aligned}
C_k &= \{ab, ac, bc, de, df, dg, ef, eg, fg\} \\
N_k &= \{ad, ae, af, ag, bd, be, bf, bg, cd, ce, cf, cg\} \\
C_r &= \{ab, cd, fg, fh, fi, gh, gi, hi\} \\
N_r &= \{ac, ad, af, ag, ah, ai, bc, bd, bf, bg, bh, bi, \\
&\quad cf, cg, ch, ci, df, dg, dh, di\}.
\end{aligned}$$

Recall and precision for coreference links are:

$$\begin{aligned}
R_c &= \frac{|C_k \cap C_r|}{|C_k|} = \frac{2}{9} \approx 0.22 \\
P_c &= \frac{|C_k \cap C_r|}{|C_r|} = \frac{2}{8} = 0.25
\end{aligned}$$

and the coreference F-measure, $F_c \approx 0.23$. Similarly, recall and precision for non-coreference links are:

$$\begin{aligned}
R_n &= \frac{|N_k \cap N_r|}{|N_k|} = \frac{8}{12} \approx 0.67 \\
P_n &= \frac{|N_k \cap N_r|}{|N_r|} = \frac{8}{20} = 0.40,
\end{aligned}$$

and the non-coreference F-measure, $F_n = 0.50$. So the BLANC score is $\frac{F_c + F_n}{2} \approx 0.36$.

5 Conclusion

We have cleared several misunderstandings about coreference evaluation metrics, especially when a response contains imperfect predicted mentions, and have argued against mention manipulations during coreference evaluation. These misunderstandings are caused partially by the lack of illustrative examples to show how a metric is computed on predicted mentions not aligned perfectly with key mentions. Therefore, we provide detailed steps for computing all four metrics on a representative example. Furthermore, we have a reference implementation of these metrics that has been rigorously tested and has been made available to the public as open source software. We reported new scores on the CoNLL 2011 and 2012 data sets, which can serve as the benchmarks for future research work.

Acknowledgments

This work was partially supported by grants R01LM10090 from the National Library of Medicine and IIS-1219142 from the National Science Foundation.

References

- Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of LREC*, pages 563–566.
- Jie Cai and Michael Strube. 2010. Evaluation metrics for end-to-end coreference resolution systems. In *Proceedings of SIGDIAL*, pages 28–36.
- Chen Chen and Vincent Ng. 2013. Linguistically aware coreference evaluation metrics. In *Proceedings of the Sixth IJCNLP*, pages 1366–1374, Nagoya, Japan, October.
- Nancy Chinchor and Beth Sundheim. 2003. Message understanding conference (MUC) 6. In *LDC2003T13*.
- Nancy Chinchor. 2001. Message understanding conference (MUC) 7. In *LDC2001T02*.
- George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program-tasks, data, and evaluation. In *Proceedings of LREC*.
- Lynette Hirschman and Nancy Chinchor. 1997. Coreference task definition (v3.0, 13 jul 97). In *Proceedings of the 7th Message Understanding Conference*.
- Gordana Ilic Holen. 2013. Critical reflections on evaluation practices in coreference resolution. In *Proceedings of the NAACL-HLT Student Research Workshop*, pages 1–7, Atlanta, Georgia, June.
- Daniel Jurafsky and James H. Martin. 2008. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall. Second Edition.
- Xiaoqiang Luo, Sameer Pradhan, Marta Recasens, and Eduard Hovy. 2014. An extension of BLANC to system mentions. In *Proceedings of ACL*, Baltimore, Maryland, June.
- Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of HLT-EMNLP*, pages 25–32.
- Sameer Pradhan, Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2007. OntoNotes: A Unified Relational Semantic Representation. *International Journal of Semantic Computing*, 1(4):405–419.
- Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. 2011. CoNLL-2011 shared task: Modeling unrestricted coreference in OntoNotes. In *Proceedings of CoNLL: Shared Task*, pages 1–27.
- Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of CoNLL: Shared Task*, pages 1–40.
- Altaf Rahman and Vincent Ng. 2009. Supervised models for coreference resolution. In *Proceedings of EMNLP*, pages 968–977.
- Altaf Rahman and Vincent Ng. 2011. Coreference resolution with world knowledge. In *Proceedings of ACL*, pages 814–824.
- Marta Recasens and Eduard Hovy. 2011. BLANC: Implementing the Rand Index for coreference evaluation. *Natural Language Engineering*, 17(4):485–510.
- Marta Recasens, Lluís Màrquez, Emili Sapena, M. Antònia Martí, Mariona Taulé, Véronique Hoste, Massimo Poesio, and Yannick Versley. 2010. Semeval-2010 task 1: Coreference resolution in multiple languages. In *Proceedings of SemEval*, pages 1–8.
- Marta Recasens, Marie-Catherine de Marneffe, and Chris Potts. 2013. The life and death of discourse entities: Identifying singleton mentions. In *Proceedings of NAACL-HLT*, pages 627–633.
- Veselin Stoyanov, Nathan Gilbert, Claire Cardie, and Ellen Riloff. 2009. Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *Proceedings of ACL-IJCNLP*, pages 656–664.
- William F. Styler, Steven Bethard and Sean Finan, Martha Palmer, Sameer Pradhan, Piet C de Groen, Brad Erickson, Timothy Miller, Chen Lin, Guergana Savova, and James Pustejovsky. 2014. Temporal annotation in the clinical domain. *Transactions of Computational Linguistics*, 2(April):143–154.
- Ozlem Uzuner, Andreea Bodnari, Shuying Shen, Tyler Forbush, John Pestian, and Brett R South. 2012. Evaluating the state of the art in coreference resolution for electronic medical records. *Journal of American Medical Informatics Association*, 19(5), September.
- Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model theoretic coreference scoring scheme. In *Proceedings of the 6th Message Understanding Conference*, pages 45–52.
- Ralph Weischedel, Eduard Hovy, Mitchell Marcus, Martha Palmer, Robert Belvin, Sameer Pradhan, Lance Ramshaw, and Nianwen Xue. 2011. OntoNotes: A large training corpus for enhanced processing. In Joseph Olive, Caitlin Christianson, and John McCary, editors, *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*. Springer.

Measuring Sentiment Annotation Complexity of Text

Aditya Joshi^{1,2,3*} Abhijit Mishra¹ Nivvedan Senthamilselvan¹
Pushpak Bhattacharyya¹

¹IIT Bombay, India, ²Monash University, Australia

³IITB-Monash Research Academy, India

{adityaj, abhijitmishra, nivvedan, pb}@cse.iitb.ac.in

Abstract

The effort required for a human annotator to detect sentiment is not uniform for all texts, irrespective of his/her expertise. We aim to predict a score that quantifies this effort, using linguistic properties of the text. Our proposed metric is called *Sentiment Annotation Complexity (SAC)*. As for training data, since any direct judgment of complexity by a human annotator is fraught with subjectivity, we rely on cognitive evidence from eye-tracking. The sentences in our dataset are labeled with SAC scores derived from *eye-fixation duration*. Using linguistic features and annotated SACs, we train a regressor that *predicts the SAC* with a best mean error rate of 22.02% for five-fold cross-validation. We also study the correlation between a human annotator’s perception of complexity and a machine’s confidence in polarity determination. The merit of our work lies in (a) deciding the sentiment annotation cost in, for example, a crowdsourcing setting, (b) choosing the right classifier for sentiment prediction.

1 Introduction

The effort required by a human annotator to detect sentiment is not uniform for all texts. Compare the hypothetical tweet “*Just what I wanted: a good pizza.*” with “*Just what I wanted: a cold pizza.*”. The two are lexically and structurally similar. However, because of the sarcasm in the second tweet (in “cold” pizza, an undesirable situation followed by a positive sentiment phrase “just what I wanted”, as discussed in Riloff et al. (2013)), it is more complex than the first for sentiment annotation. Thus, independent of how good

* Aditya is funded by the TCS Research Fellowship Program.

the annotator is, there are sentences which will be perceived to be more complex than others. With regard to this, we introduce a metric called *sentiment annotation complexity (SAC)*. The SAC of a given piece of text (sentences, in our case) can be predicted using the linguistic properties of the text as features.

The primary question is whether such complexity measurement is necessary at all. Fort et al (2012) describe the necessity of annotation complexity measurement in manual annotation tasks. Measuring annotation complexity is beneficial in annotation crowdsourcing. If the complexity of the text can be estimated *even before the annotation begins*, the pricing model can be fine-tuned (pay less for sentences that are easy to annotate, for example). Also, in terms of an automatic SA engine which has multiple classifiers in its ensemble, a classifier may be chosen based on the complexity of sentiment annotation (for example, use a rule-based classifier for simple sentences and a more complex classifier for other sentences). Our metric adds value to sentiment annotation and sentiment analysis, in these two ways. The fact that sentiment expression may be complex is evident from a study of comparative sentences by Ganapathibhotla and Liu (2008), sarcasm by Riloff et al. (2013), thwarting by Ramteke et al. (2013) or implicit sentiment by Balahur et al. (2011). To the best of our knowledge, there is no general approach to “measure” how complex a piece of text is, in terms of sentiment annotation.

The central challenge here is to annotate a data set with SAC. To measure the “actual” time spent by an annotator on a piece of text, we use an eye-tracker to record eye-fixation duration: the time for which the annotator has actually focused on the sentence during annotation. Eye-tracking annotations have been used to study the cognitive aspects of language processing tasks like translation by Dragsted (2010) and sense disambiguation by

Joshi et al. (2011). Mishra et al. (2013) present a technique to determine translation difficulty index. The work closest to ours is by Scott et al. (2011) who use eye-tracking to study the role of emotion words in reading.

The novelty of our work is three-fold: (a) *The proposition of a metric to measure complexity of sentiment annotation*, (b) *The adaptation of past work that uses eye-tracking for NLP in the context of sentiment annotation*, (c) *The learning of regressors that automatically predict SAC using linguistic features*.

2 Understanding Sentiment Annotation Complexity

The process of sentiment annotation consists of two sub-processes: comprehension (where the annotator understands the content) and sentiment judgment (where the annotator identifies the sentiment). The complexity in sentiment annotation stems from an interplay of the two and we expect SAC to capture the combined complexity of both the sub-processes. In this section, we describe how complexity may be introduced in sentiment annotation in different classical layers of NLP.

The simplest form of sentiment annotation complexity is at the **lexical level**. Consider the sentence “*It is messy, uncouth, incomprehensible, vicious and absurd*”. The sentiment words used in this sentence are uncommon, resulting in complexity.

The next level of sentiment annotation complexity arises due to **syntactic complexity**. Consider the review: “*A somewhat crudely constructed but gripping, questing look at a person so racked with self-loathing, he becomes an enemy to his own race*”. An annotator will face difficulty in comprehension as well as sentiment judgment due to the complicated phrasal structure in this review. Implicit expression of sentiment introduces complexity at the **semantic and pragmatic** level. Sarcasm expressed in “*It’s like an all-star salute to disney’s cheesy commercialism*” leads to difficulty in sentiment annotation because of positive words like “*an all-star salute*”.

Manual annotation of complexity scores may not be intuitive and reliable. Hence, we use a cognitive technique to create our annotated dataset. The underlying idea is: *if we monitor annotation of two textual units of equal length, the more complex unit will take longer to annotate, and hence,*

should have a higher SAC. Using the idea of “annotation time” linked with complexity, we devise a technique to create a dataset annotated with SAC.

It may be thought that *inter-annotator agreement (IAA)* provides implicit annotation: the higher the agreement, the easier the piece of text is for sentiment annotation. However, in case of multiple expert annotators, this agreement is expected to be high for most sentences, due to the expertise. For example, all five annotators agree with the label for 60% sentences in our data set. However, the duration for these sentences has a mean of 0.38 seconds and a standard deviation of 0.27 seconds. This indicates that although IAA is easy to compute, it does not determine sentiment annotation complexity of text in itself.

3 Creation of dataset annotated with SAC

We wish to predict sentiment annotation complexity of the text using a supervised technique. As stated above, the time-to-annotate is one good candidate. However, “simple time measurement” is not reliable because the annotator may spend time not doing any annotation due to fatigue or distraction. To accurately record the time, we use an eye-tracking device that measures the “duration of eye-fixations¹”. Another attribute recorded by the eye-tracker that may have been used is “saccade duration²”. However, saccade duration is not significant for annotation of short text, as in our case. Hence, the SAC labels of our dataset are fixation durations with appropriate normalization.

It may be noted that the eye-tracking device is used only to annotate training data. The actual prediction of SAC is done using linguistic features alone.

3.1 Eye-tracking Experimental Setup

We use a sentiment-annotated data set consisting of movie reviews by (Pang and Lee, 2005) and tweets from <http://help.sentiment140.com/for-students>. A total of 1059 sentences (566 from a movie corpus, 493 from a twitter corpus) are selected.

We then obtain two kinds of annotation from five paid annotators: (a) sentiment (positive, negative and objective), (b) eye-movement as recorded

¹A long stay of the visual gaze on a single location.

²A rapid movement of the eyes between positions of rest on the sentence.



Figure 1: Gaze-data recording using Translog-II

by an eye-tracker. They are given a set of instructions beforehand and can seek clarifications. This experiment is conducted as follows:

1. A sentence is displayed to the annotator on the screen. The annotator verbally states the sentiment of this sentence, before (s)he can proceed to the next.
2. While the annotator reads the sentence, a remote eye-tracker (Model: Tobii TX 300, Sampling rate: 300Hz) records the eye-movement data of the annotator. The eye-tracker is linked to a Translog II software (Carl, 2012) in order to record the data. A snapshot of the software is shown in figure 1. The dots and circles represent position of eyes and fixations of the annotator respectively.
3. The experiment then continues in modules of 50 sentences at a time. This is to prevent fatigue over a period of time. Thus, each annotator participates in this experiment over a number of sittings.

We ensure the quality of our dataset in different ways: (a) Our annotators are instructed to avoid unnecessary head movements and eye-movements outside the experiment environment. (b) To minimize noise due to head movements further, they are also asked to state the annotation verbally, which was then manually recorded, (c) Our annotators are students between the ages 20-24 with English as the primary language of academic instruction and have secured a TOEFL iBT score of 110 or above.

We understand that sentiment is nuanced- towards a target, through constructs like sarcasm and presence of multiple entities. However, we want to capture the most natural form of sentiment annotation. So, the guidelines are kept to a bare minimum of “*annotating a sentence as positive, negative and objective as per the speaker*”. This experiment results in a data set of 1059 sentences with

a fixation duration recorded for each sentence-annotator pair³ The multi-rater kappa IAA for sentiment annotation is 0.686.

3.2 Calculating SAC from eye-tracked data

We now need to annotate each sentence with a SAC. We extract *fixation durations* of the five annotators for each of the annotated sentences. A single SAC score for sentence s for N annotators is computed as follows:

$$SAC(s) = \frac{1}{N} \sum_{n=1}^N \frac{z(n, dur(s, n))}{len(s)} \quad (1)$$

where,

$$z(n, dur(s, n)) = \frac{dur(s, n) - \mu(dur(n))}{\sigma(dur(n))}$$

In the above formula, N is the total number of annotators while n corresponds to a specific annotator. $dur(s, n)$ is the fixation duration of annotator n on sentence s . $len(s)$ is the number of words in sentence s . This normalization over number of words assumes that long sentences may have high $dur(s, n)$ but do not necessarily have high SACs. $\mu(dur(n))$, $\sigma(dur(n))$ is the mean and standard deviation of fixation durations for annotator n across all sentences. $z(n, \cdot)$ is a function that z -normalizes the value for annotator n to standardize the deviation due to reading speeds. We convert the SAC values to a scale of 1-10 using min-max normalization. To understand how the formula records sentiment annotation complexity, consider the SACs of examples in section 2. The sentence “it is messy , uncouth , incomprehensible , vicious and absurd” has a SAC of 3.3. On the other hand, the SAC for the sarcastic sentence “it’s like an all-star salute to disney’s cheesy commercialism.” is 8.3.

4 Predictive Framework for SAC

The previous section shows how gold labels for SAC can be obtained using eye-tracking experiments. This section describes our predictive for SAC that uses four categories of linguistic features: *lexical*, *syntactic*, *semantic* and *sentiment-related* in order to capture the subprocesses of annotation as described in section 2.

4.1 Experiment Setup

The linguistic features described in Table 3.2 are extracted from the input sentences. Some of these

³The complete eye-tracking data is available at: <http://www.cfilt.iitb.ac.in/~cognitive-nlp/>.

Feature	Description
Lexical	
- Word Count	
- Degree of polysemy	Average number of Wordnet senses per word
- Mean Word Length	Average number of characters per word (commonly used in readability studies as in the case of Pascual et al. (2005))
- %ge of nouns and adjs.	
- %ge of Out-of-vocabulary words	
Syntactic	
- Dependency Distance	Average distance of all pairs of dependent words in the sentence (Lin, 1996)
- Non-terminal to Terminal ratio	Ratio of the number of non-terminals to the number of terminals in the constituency parse of a sentence
Semantic	
- Discourse connectors	Number of discourse connectors
- Co-reference distance	Sum of token distance between co-referring entities of anaphora in a sentence
- Perplexity	Trigram perplexity using language models trained on a mixture of sentences from the Brown corpus, the Amazon Movie corpus and Stanford twitter corpus (mentioned in Sections 3 and 5)
Sentiment-related (Computed using SentiWordNet (Esuli et al., 2006))	
- Subjective Word Count	
- Subjective Score	Sum of SentiWordNet scores of all words
- Sentiment Flip Count	A positive word followed in sequence by a negative word, or vice versa counts as one sentiment flip

Table 1: Linguistic Features for the Predictive Framework

features are extracted using Stanford Core NLP ⁴ tools and NLTK (Bird et al., 2009). Words that do not appear in Academic Word List ⁵ and General Service List ⁶ are treated as out-of-vocabulary words. The training data consists of 1059 tuples, with 13 features and gold labels from eye-tracking experiments.

To predict SAC, we use Support Vector Regression (SVR) (Joachims, 2006). Since we do not have any information about the nature of the relationship between the features and SAC, choosing SVR allows us to try multiple kernels. We carry out a 5-fold cross validation for both in-domain and cross-domain settings, to validate that the regressor does not overfit. The model thus learned is evaluated using: (a) Error metrics namely, Mean Squared Error estimate, Mean Absolute Error estimate and Mean Percentage Error. (b) the Pearson correlation coefficient between the gold and pre-

⁴<http://nlp.stanford.edu/software/corenlp.shtml>

⁵www.victoria.ac.nz/lals/resources/academicwordlist/

⁶www.jbauman.com/gsl.html

dicted SAC.

4.2 Results

The results are tabulated in Table 2. Our observation is that a quadratic kernel performs slightly better than linear. The correlation values are positive and indicate that even if the predicted scores are not as accurate as desired, the system is capable of ranking sentences in the correct order based on their sentiment complexity. The mean percentage error (MPE) of the regressors ranges between 22-38.21%. The cross-domain MPE is higher than the rest, as expected.

To understand how each of the features performs, we conducted ablation tests by considering one feature at a time. Based on the MPE values, the best features are: Mean word length (MPE=27.54%), Degree of Polysemy (MPE=36.83%) and %ge of nouns and adjectives (MPE=38.55%). To our surprise, word count performs the worst (MPE=85.44%). This is unlike tasks like translation where length has been shown

Kernel Domain	Linear			Quadratic			Cross Domain Linear	
	Mixed	Movie	Twitter	Mixed	Movie	Twitter	Movie	Twitter
MSE	1.79	1.55	1.99	1.68	1.53	1.88	3.17	2.24
MAE	0.93	0.89	0.95	0.91	0.88	0.93	1.39	1.19
MPE	22.49%	23.8%	25.45%	22.02%	23.8%	25%	35.01%	38.21%
Correlation	0.54	0.38	0.56	0.57	0.37	0.6	0.38	0.46

Table 2: Performance of Predictive Framework for 5-fold in-domain and cross-domain validation using Mean Squared Error (MSE), Mean Absolute Error (MAE) and Mean Percentage Error (MPE) estimates and correlation with the gold labels.

to be one of the best predictors in translation difficulty (Mishra et al., 2013). We believe that for sentiment annotation, longer sentences may have more lexical clues that help detect the sentiment more easily. Note that some errors may be introduced in feature extraction due to limitations of the NLP tools.

5 Discussion

Our proposed metric measures complexity of sentiment annotation, as perceived by human annotators. It would be worthwhile to study the *human-machine correlation* to see if *what is difficult for a machine is also difficult for a human*. In other words, the goal is to show that the confidence scores of a sentiment classifier are negatively correlated with SAC.

We use three sentiment classification techniques: Naïve Bayes, MaxEnt and SVM with unigrams, bigrams and trigrams as features. The training datasets used are: a) 10000 movie reviews from Amazon Corpus (McAuley et. al, 2013) and b) 20000 tweets from the twitter corpus (same as mentioned in section 3). Using NLTK and Scikit-learn⁷ with default settings, we generate six positive/negative classifiers, for all possible combinations of the three models and two datasets.

The confidence score of a classifier⁸ for given text t is computed as follows:

$$P : \text{Probability of predicted class}$$

$$\text{Confidence}(t) = \begin{cases} P & \text{if predicted} \\ & \text{polarity is correct} \\ 1 - P & \text{otherwise} \end{cases} \quad (2)$$

⁷<http://scikit-learn.org/stable/>

⁸In case of SVM, the probability of predicted class is computed as given in Platt (1999).

Classifier (Corpus)	Correlation
Naïve Bayes (Movie)	-0.06 (73.35)
Naïve Bayes (Twitter)	-0.13 (71.18)
MaxEnt (Movie)	-0.29 (72.17)
MaxEnt (Twitter)	-0.26 (71.68)
SVM (Movie)	-0.24 (66.27)
SVM (Twitter)	-0.19 (73.15)

Table 3: Correlation between confidence of the classifiers with SAC; Numbers in parentheses indicate classifier accuracy (%)

Table 3 presents the accuracy of the classifiers along with the correlations between the confidence score and observed SAC values. MaxEnt has the highest negative correlation of -0.29 and -0.26. For both domains, we observe a weak yet negative correlation which suggests that the perception of difficulty by the classifiers are in line with that of humans, as captured through SAC.

6 Conclusion & Future Work

We presented a metric called Sentiment Annotation Complexity (SAC), a metric in SA research that has been unexplored until now. First, the process of data preparation through eye tracking, labeled with the SAC score was elaborated. Using this data set and a set of linguistic features, we trained a regression model to predict SAC. Our predictive framework for SAC resulted in a mean percentage error of 22.02%, and a moderate correlation of 0.57 between the predicted and observed SAC values. Finally, we observe a negative correlation between the classifier confidence scores and a SAC, as expected. As a future work, we would like to investigate how SAC of a test sentence can be used to choose a classifier from an ensemble, and to determine the pre-processing steps (entity-relationship extraction, for example).

References

- Balahur, Alexandra and Hermida, Jesús M and Montoyo, Andrés. 2011. Detecting implicit expressions of sentiment in text based on commonsense knowledge. *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, 53-60.
- Batali, John and Searle, John R. 1995. The Rediscovery of the Mind. *Artif. Intell.*, Vol. 77, 177-193.
- Steven Bird and Ewan Klein and Edward Loper. 2009. Natural Language Processing with Python *O'Reilly Media*.
- Carl, M. 2012. Translog-II: A Program for Recording User Activity Data for Empirical Reading and Writing Research. In *Proceedings of the Eight International Conference on Language Resources and Evaluation, European Language Resources Association*.
- Dragsted, B. 2010. 2010. Co-ordination of reading and writing processes in translation. *Contribution to Translation and Cognition*. Shreve, G. and Angelone, E.(eds.)Cognitive Science Society.
- Esuli, Andrea and Sebastiani, Fabrizio. 2006. Sentimentnet: A publicly available lexical resource for opinion mining. *Proceedings of LREC*, vol. 6, 417-422.
- Fellbaum, Christiane 1998. WordNet: An electronic lexical database. 1998. *Cambridge, MA: MIT Press*.
- Fort, Karën and Nazarenko, Adeline and Rosset, Sophie et al 2012. Modeling the complexity of manual annotation tasks: A grid of analysis *Proceedings of the International Conference on Computational Linguistics*.
- Ganapathibhotla, G and Liu, Bing. 2008. Identifying preferred entities in comparative sentences. *22nd International Conference on Computational Linguistics (COLING)*.
- González-Ibáñez, Roberto and Muresan, Smaranda and Wacholder, Nina 2011. Identifying Sarcasm in Twitter: A Closer Look. *ACL (Short Papers)* 581-586.
- Joachims, T. 2006 Training Linear SVMs in Linear Time *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*.
- Lin, D. 1996 On the structural complexity of natural language sentences. *Proceeding of the 16th International Conference on Computational Linguistics (COLING)*, pp. 729733.
- Martinez-Gómez, Pascual and Aizawa, Akiko. 2013. Diagnosing Causes of Reading Difficulty using Bayesian Networks *International Joint Conference on Natural Language Processing*, 13831391.
- McAuley, Julian John and Leskovec, Jure 2013 From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. *Proceedings of the 22nd international conference on World Wide Web*.
- Mishra, Abhijit and Bhattacharyya, Pushpak and Carl, Michael. 2013. Automatically Predicting Sentence Translation Difficulty *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 346-351.
- Narayanan, Ramanathan and Liu, Bing and Choudhary, Alok 2009. Sentiment Analysis of Conditional Sentences. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, 180-189.
- Pang, Bo and Lee, Lillian. 2008. Opinion mining and sentiment analysis *Foundations and trends in information retrieval*, vol. 2, 1-135.
- Pang, Bo and Lee, Lillian. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 115-124.
- Platt, John and others. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods *Advances in large margin classifiers*, vol. 10, 61-74.
- Ramteke, Ankit and Malu, Akshat and Bhattacharyya, Pushpak and Nath, J. Saketha 2013. Detecting Turnarounds in Sentiment Analysis: Thwarting *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 860-865.
- Riloff, Ellen and Qadir, Ashequl and Surve, Prafulla and De Silva, Lalindra and Gilbert, Nathan and Huang, Ruihong 2013. Sarcasm as Contrast between a Positive Sentiment and Negative Situation *Conference on Empirical Methods in Natural Language Processing, Seattle, USA*.
- Salil Joshi, Diptesh Kanojia and Pushpak Bhattacharyya. 2013. More than meets the eye: Study of Human Cognition in Sense Annotation. *NAACL HLT 2013, Atlanta, USA*.
- Scott G. , O Donnell P and Sereno S. 2012. Emotion Words Affect Eye Fixations During Reading. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 2012, Vol. 38, No. 3, 783-792
- Siegel, Sidney and N. J. Castellan, Jr. 1988. *Nonparametric Statistics for the Behavioral Sciences. Second edition. McGraw-Hill*.

Improving Citation Polarity Classification with Product Reviews

Charles Jochim*

IBM Research – Ireland
charlesj@ie.ibm.com

Hinrich Schütze

Center for Information & Language Processing
University of Munich

Abstract

Recent work classifying citations in scientific literature has shown that it is possible to improve classification results with extensive feature engineering. While this result confirms that citation classification is feasible, there are two drawbacks to this approach: (i) it requires a large annotated corpus for supervised classification, which in the case of scientific literature is quite expensive; and (ii) feature engineering that is too specific to one area of scientific literature may not be portable to other domains, even within scientific literature. In this paper we address these two drawbacks. First, we frame citation classification as a domain adaptation task and leverage the abundant labeled data available in other domains. Then, to avoid over-engineering specific citation features for a particular scientific domain, we explore a deep learning neural network approach that has shown to generalize well across domains using unigram and bigram features. We achieve better citation classification results with this cross-domain approach than using in-domain classification.

1 Introduction

Citations have been categorized and studied for a half-century (Garfield, 1955) to better understand when and how citations are used, and to record and measure how information is exchanged (e.g., networks of co-cited papers or authors (Small and Griffith, 1974)). Recently, the value of this information has been shown in practical applications such as information retrieval (IR)

*This work was primarily conducted at the IMS – University of Stuttgart.

(Ritchie et al., 2008), summarization (Qazvinian and Radev, 2008), and even identifying scientific breakthroughs (Small and Klavans, 2011). We expect that by identifying and labeling the *function* of citations we can improve the effectiveness of these applications.

There has been no consensus on what aspects or functions of a citation should be annotated and how. Early citation classification focused more on *citation motivation* (Garfield, 1964), while later classification considered more the *citation function* (Chubin and Moitra, 1975). Recent studies using automatic classification have continued this tradition of introducing a new classification scheme with each new investigation into the use of citations (Nanba and Okumura, 1999; Teufel et al., 2006a; Dong and Schäfer, 2011; Abu-Jbara et al., 2013). One distinction that has been more consistently annotated across recent citation classification studies is between *positive* and *negative* citations (Athar, 2011; Athar and Teufel, 2012; Abu-Jbara et al., 2013).¹ The popularity of this distinction likely owes to the prominence of sentiment analysis in NLP (Liu, 2010). We follow much of the recent work on citation classification and concentrate on citation polarity.

2 Domain Adaptation

By concentrating on citation polarity we are able to compare our classification to previous citation polarity work. This choice also allows us to access the wealth of existing data containing polarity annotation and then frame the task as a domain adaptation problem. Of course the risk in approaching the problem as domain adaptation is that the domains are so different that the representation of a positive instance of a movie or product review, for example, will not coincide with that of a posi-

¹Dong and Schäfer (2011) also annotate polarity, which can be found in their dataset (described later), but this is not discussed in their paper.

tive scientific citation. On the other hand, because there is a limited amount of annotated citation data available, by leveraging large amounts of annotated polarity data we could potentially even improve citation classification.

We treat citation polarity classification as a sentiment analysis domain adaptation task and therefore must be careful not to define features that are too domain specific. Previous work in citation polarity classification focuses on finding new citation features to improve classification, borrowing a few from text classification in general (e.g., *n*-grams), and perhaps others from sentiment analysis problems (e.g., the polarity lexicon from Wilson et al. (2005)). We would like to do as little feature engineering as possible to ensure that the features we use are meaningful across domains. However, we do still want features that somehow capture the inherent positivity or negativity of our labeled instances, i.e., citations or Amazon product reviews. Currently a popular approach for accomplishing this is to use deep learning neural networks (Bengio, 2009), which have been shown to perform well on a variety of NLP tasks using only bag-of-word features (Collobert et al., 2011). More specifically related to our work, deep learning neural networks have been successfully employed for sentiment analysis (Socher et al., 2011) and for sentiment domain adaptation (Glorot et al., 2011). In this paper we examine one of these approaches, marginalized stacked denoising autoencoders (mSDA) from Chen et al. (2012), which has been successful in classifying the polarity of Amazon product reviews across product domains. Since mSDA achieved state-of-the-art performance in Amazon product domain adaptation, we are hopeful it will also be effective when switching to a more distant domain like scientific citations.

3 Experimental Setup

3.1 Corpora

We are interested in domain adaptation for citation classification and therefore need a target dataset of citations and a non-citation source dataset. There are two corpora available that contain citation function annotation, the DFKI Citation Corpus (Dong and Schäfer, 2011) and the IMS Citation Corpus (Jochim and Schütze, 2012). Both corpora have only about 2000 instances; unfortunately, there are no larger corpora available with citation

annotation and this task would benefit from more annotated data. Due to the infrequent use of negative citations, a substantial annotation effort (annotating over 5 times more data) would be necessary to reach 1000 negative citation instances, which is the number of negative instances in a single domain in the multi-domain corpus described below.

The DFKI Citation Corpus² has been used for classifying citation function (Dong and Schäfer, 2011), but the dataset also includes polarity annotation. The dataset has 1768 citation sentences with polarity annotation: 190 are labeled as *positive*, 57 as *negative*, and the vast majority, 1521, are left *neutral*. The second citation corpus, the IMS Citation Corpus³ contains 2008 annotated citations: 1836 are labeled *positive* and 172 are labeled *negative*. Jochim and Schütze (2012) use annotation labels from Moravcsik and Murugesan (1975) where positive instances are labeled *confirmative*, negative instances are labeled *negational*, and there is no neutral class. Because each of the citation corpora is of modest size we combine them to form one citation dataset, which we will refer to as CITD. The two citation corpora comprising CITD both come from the ACL Anthology (Bird et al., 2008): the IMS corpus uses the ACL proceedings from 2004 and the DFKI corpus uses parts of the proceedings from 2007 and 2008. Since mSDA also makes use of large amounts of unlabeled data, we extend our CITD corpus with citations from the proceedings of the remaining years of the ACL, 1979–2003, 2005–2006, and 2009.

There are a number of non-citation corpora available that contain polarity annotation. For these experiments we use the Multi-Domain Sentiment Dataset⁴ (henceforth MDSD), introduced by Blitzer et al. (2007). We use the version of the MDSD that includes *positive* and *negative* labels for product reviews taken from Amazon.com in the following domains: books, dvd, electronics, and kitchen. For each domain there are 1000 positive reviews and 1000 negative reviews that comprise the “labeled” data, and then roughly 4000 more reviews in the “unlabeled”⁵ data. Reviews

²https://aclbib.opendfki.de/repos/trunk/citation_classification_dataset/

³<http://www.ims.uni-stuttgart.de/~jochimcs/citation-classification/>

⁴<http://www.cs.jhu.edu/~mdredze/datasets/sentiment/>

⁵It is usually treated as unlabeled data even though it ac-

Corpus	Instances	Pos.	Neg.	Neut.
DFKI	1768	190	57	1521
IMS	2008	1836	172	–
MDSO	27,677	13,882	13,795	–

Table 1: Polarity corpora.

were preprocessed so that for each review you find a list of unigrams and bigrams with their frequency within the review. Unigrams from a stop list of 55 stop words are removed, but stop words in bigrams remain.

Table 1 shows the distribution of polarity labels in the corpora we use for our experiments. We combine the DFKI and IMS corpora into the CITD corpus. We omit the citations labeled *neutral* from the DFKI corpus because the IMS corpus does not contain neutral annotation nor does the MDSO. It is the case in many sentiment analysis corpora that only positive and negative instances are included, e.g., (Pang et al., 2002).

The citation corpora presented above are both unbalanced and both have a highly skewed distribution. The MDSO on the other hand is evenly balanced and an effort was even made to keep the data treated as “unlabeled” rather balanced. For this reason, in line with previous work using MDSO, we balance the labeled portion of the CITD corpus. This is done by taking 179 unique negative sentences in the DFKI and IMS corpora and randomly selecting an equal number of positive sentences. The IMS corpus can have multiple labeled citations per sentence: there are 122 sentences containing the 172 negative citations from Table 1. The final CITD corpus comprises this balanced corpus of 358 labeled citation sentences plus another 22,093 unlabeled citation sentences.

3.2 Features

In our experiments, we restrict our features to unigrams and bigrams from the product review or citation context (i.e., the sentence containing the citation). This follows previous studies in domain adaptation (Blitzer et al., 2007; Glorot et al., 2011). Chen et al. (2012) achieve state-of-the-art results on MDSO by testing the 5000 and 30,000 most frequent unigram and bigram features.

Previous work in citation classification has largely focused on identifying new features for

improving classification accuracy. A significant amount of effort goes into engineering new features, in particular for identifying cue phrases, e.g., (Teufel et al., 2006b; Dong and Schäfer, 2011). However, there seems to be little consensus on which features help most for this task. For example, Abu-Jbara et al. (2013) and Jochim and Schütze (2012) find the list of polar words from Wilson et al. (2005) to be useful, and neither study lists dependency relations as significant features. Athar (2011) on the other hand reported significant improvement using dependency relation features and found that the same list of polar words slightly hurt classification accuracy. The classifiers and implementation of features varies between these studies, but the problem remains that there seems to be no clear set of features for citation polarity classification.

The lack of consensus on the most useful citation polarity features coupled with the recent success of deep learning neural networks (Collobert et al., 2011) further motivate our choice to limit our features to the n -grams available in the product review or citation context and not rely on external resources or tools for additional features.

3.3 Classification with mSDA

For classification we use marginalized stacked denoising autoencoders (mSDA) from Chen et al. (2012)⁶ plus a linear SVM. mSDA takes the concept of *denoising* – introducing noise to make the autoencoder more robust – from Vincent et al. (2008), but does the optimization in closed form, thereby avoiding iterating over the input vector to stochastically introduce noise. The result of this is faster run times and currently state-of-the-art performance on MDSO, which makes it a good choice for our domain adaptation task. The mSDA implementation comes with LIBSVM, which we replace with LIBLINEAR (Fan et al., 2008) for faster run times with no decrease in accuracy. LIBLINEAR, with default settings, also serves as our baseline.

3.4 Outline of Experiments

Our initial experiments simply extend those of Chen et al. (2012) (and others who have used MDSO) by adding another domain, citations. We train on each of the domains from the MDSO –

⁶We use their MATLAB implementation available at <http://www.cse.wustl.edu/~mchen/code/mSDA.tar>.

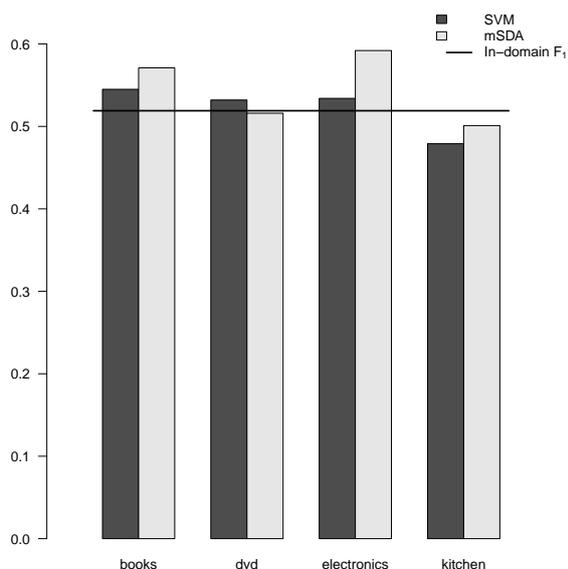


Figure 1: Cross domain macro- F_1 results training on Multi-Domain Sentiment Dataset and testing on citation dataset (CITD). The horizontal line indicates macro- F_1 for in-domain citation classification.

books, dvd, electronics, and kitchen – and test on the citation data. We split the labeled data 80/20 following Blitzer et al. (2007) (cf. Chen et al. (2012) train on all “labeled” data and test on the “unlabeled” data). These experiments should help answer two questions: does a larger amount of training data, even if out of domain, improve citation classification; and how well do the different product domains generalize to citations (i.e., which domains are most similar to citations)?

In contrast to previous work using MDS, a lot of the work in domain adaptation also leverages a small amount of labeled target data. In our second set of experiments, we follow the domain adaptation approaches described in (Daumé III, 2007) and train on product review and citation data before testing on citations.

4 Results and Discussion

4.1 Citation mSDA

Our initial results show that using mSDA for domain adaptation to citations actually outperforms in-domain classification. In Figure 1 we compare citation classification with mSDA to the SVM baseline. Each pair of vertical bars represents training on a domain from MDS (e.g., books) and testing on CITD. The dark gray bar indicates the F_1 scores for the SVM baseline using the

30,000 features and the lighter gray bar shows the mSDA results. The black horizontal line indicates the F_1 score for in-domain citation classification, which sometimes represents the goal for domain adaptation. We can see that using a larger dataset, even if out of domain, does improve citation classification. For books, dvd, and electronics, even the SVM baseline improves on in-domain classification. mSDA does better than the baseline for all domains except dvd. Using a larger training set, along with mSDA, which makes use of the unlabeled data, leads to the best results for citation classification.

In domain adaptation we would expect the domains most similar to the target to lead to the highest results. Like Dai et al. (2007), we measure the Kullback-Leibler divergence between the source and target domains’ distributions. According to this measure, citations are most similar to the books domain. Therefore, it is not surprising that training on books performs well on citations, and intuitively, among the domains in the Amazon dataset, a book review is most similar to a scientific citation. This makes the good mSDA results for electronics a bit more surprising.

4.2 Easy Domain Adaptation

The results in Section 4.1 are for *semi-supervised* domain adaptation: the case where we have some large annotated corpus (Amazon product reviews) and a large unannotated corpus (citations). There have been a number of other successful attempts at *fully supervised* domain adaptation, where it is assumed that some small amount of data is annotated in the target domain (Chelba and Acero, 2004; Daumé III, 2007; Jiang and Zhai, 2007). To see how mSDA compares to supervised domain adaptation we take the various approaches presented by Daumé III (2007). The results of this comparison can be seen in Table 2. Briefly, “All” trains on source and target data; “Weight” is the same as “All” except that instances may be weighted differently based on their domain (weights are chosen on a development set); “Pred” trains on the source data, makes predictions on the target data, and then trains on the target data with the predictions; “LinInt” linearly interpolates predictions using the source-only and target-only models (the interpolation parameter is chosen on a development set); “Augment” uses a larger feature set with source-specific and target-specific copies of features; see

Domain	Baseline	All	Weight	Pred	LinInt	Augment	mSDA
books	54.5	54.8	52.0	51.9	53.4	53.4	57.1
dvd	53.2	50.9	56.0	53.4	51.9	47.5	51.6
electronics	53.4	49.0	50.5	53.4	54.8	51.9	59.2
kitchen	47.9	48.8	50.7	53.4	52.6	49.2	50.1
citations	51.9	–	–	–	–	–	54.9

Table 2: Macro- F_1 results on CITD using different domain adaptation approaches.

(Daumé III, 2007) for further details.

We are only interested in citations as the target domain. Daumé’s source-only baseline corresponds to the “Baseline” column for domains: books, dvd, electronics, and kitchen; while his target-only baseline can be seen for citations in the last row of the “Baseline” column in Table 2.

The semi-supervised mSDA performs quite well with respect to the fully supervised approaches, obtaining the best results for books and electronics, which are also the highest scores overall. Weight and Pred have the highest F_1 scores for dvd and kitchen respectively. Daumé III (2007) noted that the “Augment” algorithm performed best when the target-only results were better than the source-only results. When this was not the case in his experiments, i.e., for the treebank chunking task, both Weight and Pred were among the best approaches. In our experiments, training on source-only outperforms target-only, with the exception of the kitchen domain.

We have included the line for citations to see the results training only on the target data ($F_1 = 51.9$) and to see the improvement when using all of the unlabeled data with mSDA ($F_1 = 54.9$).

4.3 Discussion

These results are very promising. Although they are not quite as high as other published results for citation polarity (Abu-Jbara et al., 2013)⁷, we have shown that you can improve citation polarity classification by leveraging large amounts of annotated data from other domains and using a simple set of features.

mSDA and fully supervised approaches can also be straightforwardly combined. We do not present those results here due to space constraints. The

⁷Their work included a CRF model to identify the citation context that gave them an increase of 9.2 percent F_1 over a single sentence citation context. Our approach achieves similar macro- F_1 on only the citation sentence, but using a different corpus.

combination led to mixed results: adding mSDA to the supervised approaches tended to improve F_1 over those approaches but results never exceeded the top mSDA numbers in Table 2.

5 Related Work

Teufel et al. (2006b) introduced automatic citation function classification, with classes that could be grouped as positive, negative, and neutral. They relied in part on a manually compiled list of cue phrases that cannot easily be transferred to other classification schemes or other scientific domains. Athar (2011) followed this and was the first to specifically target polarity classification on scientific citations. He found that dependency tuples contributed the most significant improvement in results. Abu-Jbara et al. (2013) also looks at both citation function and citation polarity. A big contribution of this work is that they also train a CRF sequence tagger to find the citation context, which significantly improves results over using only the citing sentence. Their feature analysis indicates that lexicons for negation, speculation, and polarity were most important for improving polarity classification.

6 Conclusion

Robust citation classification has been hindered by the relative lack of annotated data. In this paper we successfully use a large, out-of-domain, annotated corpus to improve the citation polarity classification. Our approach uses a deep learning neural network for domain adaptation with labeled out-of-domain data and unlabeled in-domain data. This semi-supervised domain adaptation approach outperforms the in-domain citation polarity classification and other fully supervised domain adaptation approaches.

Acknowledgments. We thank the DFG for funding this work (SPP 1335 *Scalable Visual Analytics*).

References

- Amjad Abu-Jbara, Jefferson Ezra, and Dragomir Radev. 2013. Purpose and polarity of citation: Towards NLP-based bibliometrics. In *Proceedings of NAACL-HLT*, pages 596–606.
- Awais Athar and Simone Teufel. 2012. Context-enhanced citation sentiment detection. In *Proceedings of NAACL-HLT*, pages 597–601.
- Awais Athar. 2011. Sentiment analysis of citations using sentence structure-based features. In *Proceedings of ACL Student Session*, pages 81–87.
- Yoshua Bengio. 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127.
- Steven Bird, Robert Dale, Bonnie Dorr, Bryan Gibson, Mark Joseph, Min-Yen Kan, Dongwon Lee, Brett Powley, Dragomir Radev, and Yee Fan Tan. 2008. The ACL anthology reference corpus: A reference dataset for bibliographic research in computational linguistics. In *Proceedings of LREC*, pages 1755–1759.
- John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL*, pages 440–447.
- Ciprian Chelba and Alex Acero. 2004. Adaptation of maximum entropy capitalizer: Little data can help a lot. In *Proceedings of EMNLP*, pages 285–292.
- Minmin Chen, Zhixiang Eddie Xu, Kilian Q. Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of ICML*, pages 767–774.
- Daryl E. Chubin and Soumyo D. Moitra. 1975. Content analysis of references: Adjunct or alternative to citation counting? *Social Studies of Science*, 5:423–441.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.
- Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. 2007. Transferring naive bayes classifiers for text classification. In *AAAI*, pages 540–545.
- Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of ACL*, pages 256–263.
- Cailing Dong and Ulrich Schäfer. 2011. Ensemble-style self-training on citation classification. In *Proceedings of IJCNLP*, pages 623–631.
- Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.
- Eugene Garfield. 1955. Citation indexes to science: A new dimension in documentation through association of ideas. *Science*, 122:108–111.
- Eugene Garfield. 1964. Can citation indexing be automated? In *Statistical Association Methods for Mechanized Documentation, Symposium Proceedings*, pages 189–192.
- Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of ICML*, pages 513–520.
- Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *ACL*, pages 264–271.
- Charles Jochim and Hinrich Schütze. 2012. Towards a generic and flexible citation classifier based on a faceted classification scheme. In *Proceedings of COLING*, pages 1343–1358.
- Bing Liu. 2010. Sentiment analysis and subjectivity. In Nitin Indurkha and Fred J. Damerau, editors, *Handbook of Natural Language Processing, Second Edition*. CRC Press, Taylor and Francis Group.
- Michael J. Moravcsik and Poovanalngam Murugesan. 1975. Some results on the function and quality of citations. *Social Studies of Science*, 5:86–92.
- Hidetsugu Nanba and Manabu Okumura. 1999. Towards multi-paper summarization using reference information. In *Proceedings of IJCAI*, pages 926–931.
- Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP*, pages 79–86.
- Vahed Qazvinian and Dragomir R. Radev. 2008. Scientific paper summarization using citation summary networks. In *Proceedings of COLING*, pages 689–696.
- Anna Ritchie, Stephen Robertson, and Simone Teufel. 2008. Comparing citation contexts for information retrieval. In *Proceedings of CIKM*, pages 213–222.
- Henry G. Small and Belder C. Griffith. 1974. The structure of scientific literatures I: Identifying and graphing specialties. *Science Studies*, 4(1):17–40.
- Henry Small and Richard Klavans. 2011. Identifying scientific breakthroughs by combining co-citation analysis and citation context. In *Proceedings of International Society for Scientometrics and Informetrics*.
- Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of EMNLP*, pages 151–161.

- Simone Teufel, Advaith Siddharthan, and Dan Tidhar.
2006a. An annotation scheme for citation function.
In *Proceedings of SIGdial Workshop on Discourse
and Dialogue*, pages 80–87.
- Simone Teufel, Advaith Siddharthan, and Dan Tidhar.
2006b. Automatic classification of citation function.
In *Proceedings of EMNLP*, pages 103–110.
- Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and
Pierre-Antoine Manzagol. 2008. Extracting and
composing robust features with denoising autoen-
coders. In *Proceedings of ICML*, pages 1096–1103.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann.
2005. Recognizing contextual polarity in phrase-
level sentiment analysis. In *Proceedings of HLT-
EMNLP*, pages 347–354.

Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification

Li Dong^{†*} Furu Wei[‡] Chuanqi Tan^{†*} Duyu Tang^{†*} Ming Zhou[‡] Ke Xu[†]

[†]Beihang University, Beijing, China

[‡]Microsoft Research, Beijing, China

[†]Harbin Institute of Technology, Harbin, China

donglixp@gmail.com fuwei@microsoft.com {ysjtcq,tangduyu}@gmail.com
mingzhou@microsoft.com kexu@nlsde.buaa.edu.cn

Abstract

We propose Adaptive Recursive Neural Network (AdaRNN) for target-dependent Twitter sentiment classification. AdaRNN adaptively propagates the sentiments of words to target depending on the context and syntactic relationships between them. It consists of more than one composition functions, and we model the adaptive sentiment propagations as distributions over these composition functions. The experimental studies illustrate that AdaRNN improves the baseline methods. Furthermore, we introduce a manually annotated dataset for target-dependent Twitter sentiment analysis.

1 Introduction

Twitter becomes one of the most popular social networking sites, which allows the users to read and post messages (i.e. tweets) up to 140 characters. Among the great varieties of topics, people in Twitter tend to express their opinions for the brands, celebrities, products and public events. As a result, it attracts much attention to estimate the crowd's sentiments in Twitter.

For the tweets, our task is to classify their sentiments for a given target as positive, negative, and neutral. People may mention several entities (or targets) in one tweet, which affects the availabilities for most of existing methods. For example, the tweet “@ballmer: windows phone is better than ios!” has three targets (@ballmer, windows phone, and ios). The user expresses neutral, positive, and negative sentiments for them, respectively. If target information is ignored, it is difficult to obtain the correct sentiment for a specified target. For target-dependent sentiment classification, the manual evaluation of Jiang et al. (2011)

show that about 40% of errors are caused by not considering the targets in classification.

The features used in traditional learning-based methods (Pang et al., 2002; Nakagawa et al., 2010) are independent to the targets, hence the results are computed despite what the targets are. Hu and Liu (2004) regard the features of products as targets, and sentiments for them are heuristically determined by the dominant opinion words. Jiang et al. (2011) combine the target-independent features (content and lexicon) and target-dependent features (rules based on the dependency parsing results) together in subjectivity classification and polarity classification for tweets.

In this paper, we mainly focus on integrating target information with Recursive Neural Network (RNN) to leverage the ability of deep learning models. The neural models use distributed representation (Hinton, 1986; Rumelhart et al., 1986; Bengio et al., 2003) to automatically learn features for target-dependent sentiment classification. RNN utilizes the recursive structure of text, and it has achieved state-of-the-art sentiment analysis results for movie review dataset (Socher et al., 2012; Socher et al., 2013). The recursive neural models employ the semantic composition functions, which enables them to handle the complex compositionality in sentiment analysis.

Specifically, we propose a framework which learns to propagate the sentiments of words towards the target depending on context and syntactic structure. We employ a novel adaptive multi-compositionality layer in recursive neural network, which is named as *AdaRNN* (Dong et al., 2014). It consists of more than one composition functions, and we model the adaptive sentiment propagations as learning distributions over these composition functions. We automatically learn the composition functions and how to select them from supervisions, instead of choosing them heuristically or by hand-crafted rules. *AdaRNN*

*Contribution during internship at Microsoft Research.

determines how to propagate the sentiments towards the target and handles the negation or intensification phenomena (Taboada et al., 2011) in sentiment analysis. In addition, we introduce a manually annotated dataset, and conduct extensive experiments on it. The experimental results suggest that our approach yields better performances than the baseline methods.

2 RNN: Recursive Neural Network

RNN (Socher et al., 2011) represents the phrases and words as D -dimensional vectors. It performs compositions based on the binary trees, and obtain the vector representations in a bottom-up way.

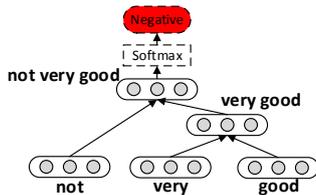


Figure 1: The composition process for “not very good” in Recursive Neural Network.

As illustrated in Figure 1, we obtain the representation of “very good” by the composition of “very” and “good”, and the representation of tri-gram “not very good” is recursively obtained by the vectors of “not” and “very good”. The dimensions of parent node are calculated by linear combination of the child vectors’ dimensions. The vector representation \mathbf{v} is obtained via:

$$\mathbf{v} = f(g(\mathbf{v}_l, \mathbf{v}_r)) = f\left(\mathbf{W} \begin{bmatrix} \mathbf{v}_l \\ \mathbf{v}_r \end{bmatrix} + \mathbf{b}\right) \quad (1)$$

where $\mathbf{v}_l, \mathbf{v}_r$ are the vectors of its left and right child, g is the composition function, f is the non-linearity function (such as tanh, sigmoid, softsign, etc.), $\mathbf{W} \in \mathbb{R}^{D \times 2D}$ is the composition matrix, and \mathbf{b} is the bias vector. The dimension of \mathbf{v} is the same as its child vectors, and it is recursively used in the next step. Notably, the word vectors in the leaf nodes are regarded as the parameters, and will be updated according to the supervisions.

The vector representation of root node is then fed into a softmax classifier to predict the label. The k -th element of softmax(\mathbf{x}) is $\frac{\exp\{\mathbf{x}_k\}}{\sum_j \exp\{\mathbf{x}_j\}}$. For a vector, the softmax obtains the distribution over K classes. Specifically, the predicted distribution is $\mathbf{y} = \text{softmax}(\mathbf{U}\mathbf{v})$, where \mathbf{y} is the predicted distribution, $\mathbf{U} \in \mathbb{R}^{K \times D}$ is the classification matrix, and \mathbf{v} is the vector representation of node.

3 Our Approach

We use the dependency parsing results to find the words syntactically connected with the interested target. Adaptive Recursive Neural Network is proposed to propagate the sentiments of words to the target node. We model the adaptive sentiment propagations as semantic compositions. The computation process is conducted in a bottom-up manner, and the vector representations are computed recursively. After we obtain the representation of target node, a classifier is used to predict the sentiment label according to the vector.

In Section 3.1, we show how to build recursive structure for target using the dependency parsing results. In Section 3.2, we propose Adaptive Recursive Neural Network and use it for target-dependent sentiment analysis.

3.1 Build Recursive Structure

The dependency tree indicates the dependency relations between words. As described above, we propagate the sentiments of words to the target. Hence the target is placed at the root node to combine with its connected words recursively. The dependency relation types are remained to guide the sentiment propagations in our model.

Algorithm 1 Convert Dependency Tree

Input: Target node, Dependency tree

Output: Converted tree

- 1: **function** CONV(r)
 - 2: $E_r \leftarrow \text{SORT}(\text{dep edges connected with } r)$
 - 3: $v \leftarrow r$
 - 4: **for** ($r \xrightarrow{t} u/u \xrightarrow{t} r$) **in** E_r **do**
 - 5: **if** r is head of u **then**
 - 6: $w \leftarrow$ node with CONV(u), v as children
 - 7: **else**
 - 8: $w \leftarrow$ node with v , CONV(u) as children
 - 9: $v \leftarrow w$
 - 10: **return** v
 - 11: Call CONV(target node) to get converted tree
-

As illustrated in the Algorithm 1, we recursively convert the dependency tree starting from the target node. We find all the words connected to the target, and these words are combined with target node by certain order. Every combination is considered as once propagation of sentiments. If the target is head of the connected words, the target vector is combined as the right node; if otherwise, it is combined as the left node. This ensures the

child nodes in a certain order. We use two rules to determine the order of combinations: (1) the words whose head is the target in dependency tree are first combined, and then the rest of connected words are combined; (2) if the first rule cannot determine the order, the connected words are sorted by their positions in sentence from right to left. Notably, the conversion is performed recursively for the connected words and the dependency relation types are remained. Figure 2 shows the converted results for different targets in one sentence.

3.2 AdaRNN: Adaptive Recursive Neural Network

RNN employs one global matrix to linearly combine the elements of vectors. Sometimes it is challenging to obtain a single powerful function to model the semantic composition, which motivates us to propose AdaRNN. The basic idea of AdaRNN is to use more than one composition functions and adaptively select them depending on the linguistic tags and the combined vectors. The model learns to propagate the sentiments of words by using the different composition functions.

Figure 2 shows the computation process for the example sentence “*windows is better than ios*”, where the user expresses positive sentiment towards *windows* and negative sentiment to *ios*. For the targets, the order of compositions and the dependency types are different. AdaRNN adaptively selects the composition functions $g_1 \dots g_C$ depending on the child vectors and the linguistic types. Thus it is able to determine how to propagate the sentiments of words towards the target.

Based on RNN described in Section 2, we define the composition result \mathbf{v} in AdaRNN as:

$$\mathbf{v} = f \left(\sum_{h=1}^C P(g_h | \mathbf{v}_l, \mathbf{v}_r, \mathbf{e}) g_h(\mathbf{v}_l, \mathbf{v}_r) \right) \quad (2)$$

where g_1, \dots, g_C are the composition functions, $P(g_h | \mathbf{v}_l, \mathbf{v}_r, \mathbf{e})$ is the probability of employing g_h given the child vectors $\mathbf{v}_l, \mathbf{v}_r$ and external feature vector \mathbf{e} , and f is the nonlinearity function. For the composition functions, we use the same forms as in Equation (1), i.e., we have C composition matrices $W_1 \dots W_C$. We define the distribution over these composition functions as:

$$\begin{bmatrix} P(g_1 | \mathbf{v}_l, \mathbf{v}_r, \mathbf{e}) \\ \vdots \\ P(g_C | \mathbf{v}_l, \mathbf{v}_r, \mathbf{e}) \end{bmatrix} = \text{softmax} \left(\beta S \begin{bmatrix} \mathbf{v}_l \\ \mathbf{v}_r \\ \mathbf{e} \end{bmatrix} \right) \quad (3)$$

where β is the hyper-parameter, $S \in \mathbb{R}^{C \times (2D+|\mathbf{e}|)}$ is the matrix used to determine which composition function we use, $\mathbf{v}_l, \mathbf{v}_r$ are the left and right child vectors, and \mathbf{e} are external feature vector. In this work, \mathbf{e} is a one-hot binary feature vector which indicates what the dependency type is. If relation is the k -th type, we set e_k to 1 and the others to 0.

Adding β in softmax function is a widely used parametrization method in statistical mechanics, which is known as Boltzmann distribution and Gibbs measure (Georgii, 2011). When $\beta = 0$, this function produces a uniform distribution; when $\beta = 1$, it is the same as softmax function; when $\beta \rightarrow \infty$, it only activates the dimension with maximum weight, and sets its probability to 1.

3.3 Model Training

We use the representation of root node as the features, and feed them into the softmax classifier to predict the distribution over classes. We define the ground truth vector \mathbf{t} as a binary vector. If the k -th class is the label, only t_k is 1 and the others are 0. Our goal is to minimize the cross-entropy error between the predicted distribution \mathbf{y} and ground truth distribution \mathbf{t} . For each training instance, we define the objective function as:

$$\min_{\Theta} - \sum_j \mathbf{t}_j \log \mathbf{y}_j + \sum_{\theta \in \Theta} \lambda_{\theta} \|\theta\|_2^2 \quad (4)$$

where Θ represents the parameters, and the L_2 -regularization penalty is used.

Based on the converted tree, we employ back-propagation algorithm (Rumelhart et al., 1986) to propagate the errors from root node to the leaf nodes. We calculate the derivatives to update the parameters. The AdaGrad (Duchi et al., 2011) is employed to solve this optimization problem.

4 Experiments

As people tend to post comments for the celebrities, products, and companies, we use these keywords (such as “*bill gates*”, “*taylor swift*”, “*xbox*”, “*windows 7*”, “*google*”) to query the Twitter API. After obtaining the tweets, we manually annotate the sentiment labels (negative, neutral, positive) for these targets. In order to eliminate the effects of data imbalance problem, we randomly sample the tweets and make the data balanced. The negative, neutral, positive classes account for 25%, 50%, 25%, respectively. Training data consists of 6,248 tweets, and testing data has 692

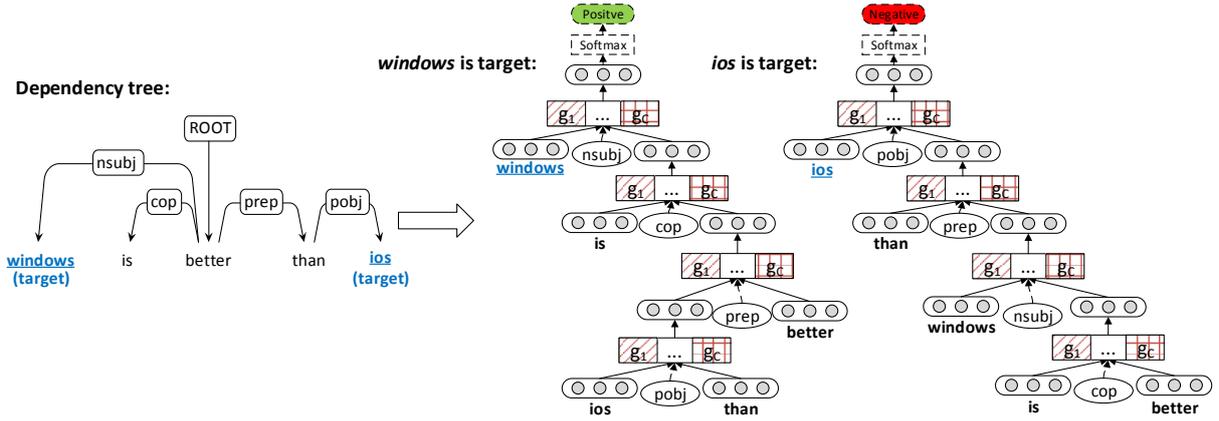


Figure 2: For the sentence “*windows is better than ios*”, we convert its dependency tree for the different targets (*windows* and *ios*). AdaRNN performs semantic compositions in bottom-up manner and forward propagates sentiment information to the target node. The g_1, \dots, g_C are different composition functions, and the combined vectors and dependency types are used to select them adaptively. These composition functions decide how to propagate the sentiments to the target.

tweets. We randomly sample some tweets, and they are assigned with sentiment labels by two annotators. About 82.5% of them have the same labels. The agreement percentage of polarity classification is higher than subjectivity classification. To the best of our knowledge, this is the largest target-dependent Twitter sentiment classification dataset which is annotated manually. We make the dataset publicly available¹ for research purposes.

We preprocess the tweets by replacing the targets with $\$T\$$ and setting their POS tags to NN . Liblinear (Fan et al., 2008) is used for baselines. A tweet-specific tokenizer (Gimpel et al., 2011) is employed, and the dependency parsing results are computed by Stanford Parser (Klein and Manning, 2003). The hyper-parameters are chosen by cross-validation on the training split, and the test accuracy and macro-average F1-score score are reported. For recursive neural models, the dimension of word vector is set to 25, and $f = \tanh$ is used as the nonlinearity function. We employ 10 composition matrices in AdaRNN. The parameters are randomly initialized. Notably, the word vectors will also be updated.

SVM-indep: It uses the uni-gram, bi-gram, punctuations, emoticons, and #hashtags as the content features, and the numbers of positive or negative words in General Inquirer as lexicon features. These features are all target-independent.

SVM-dep: We re-implement the method proposed by Jiang et al. (2011). It combines both

the target-independent (SVM-indep) and target-dependent features and uses SVM as the classifier. There are seven rules to extract target-sensitive features. We do not implement the social graph optimization and target expansion tricks in it.

SVM-conn: The words, punctuations, emoticons, and #hashtags included in the converted dependency tree are used as the features for SVM.

RNN: It is performed on the converted dependency tree without adaptive composition selection.

AdaRNN-w/oE: Our approach without using the dependency types as features in adaptive selection for the composition functions.

AdaRNN-w/E: Our approach with employing the dependency types as features in adaptive selection for the composition functions.

AdaRNN-comb: We combine the root vectors obtained by AdaRNN-w/E with the uni/bi-gram features, and they are fed into a SVM classifier.

Method	Accuracy	Macro-F1
SVM-indep	62.7	60.2
SVM-dep	63.4	63.3
SVM-conn	60.0	59.6
RNN	63.0	62.8
AdaRNN-w/oE	64.9	64.4
AdaRNN-w/E	65.8	65.5
AdaRNN-comb	66.3	65.9

Table 1: Evaluation results on target-dependent Twitter sentiment classification dataset. Our approach outperforms the baseline methods.

¹<http://goo.gl/5Enpu7>

As shown in the Table 1, AdaRNN achieves better results than the baselines. Specifically, we find that the performances of SVM-dep increase than SVM-indep. It indicates that target-dependent features help improve the results. However, the accuracy and F1-score do not gain significantly. This is caused by mismatch of the rules (Jiang et al., 2011) used to extract the target-dependent features. The POS tagging and dependency parsing results are not precise enough for the Twitter data, so these hand-crafted rules are rarely matched. Further, the results of SVM-conn illustrate that using the words which have paths to target as bag-of-words features does not perform well.

RNN is also based on the converted dependency tree. It outperforms SVM-indep, and is comparable with SVM-dep. The performances of AdaRNN-w/oE are better than the above baselines. It shows that multiple composition functions and adaptive selection help improve the results. AdaRNN provides more powerful composition ability, so that it achieves better semantic composition for recursive neural models. AdaRNN-w/E obtains best performances among the above methods. Its macro-average F1-score rises by 5.3% than the target-independent method SVM-indep. It employs dependency types as binary features to select the composition functions adaptively. The results illustrate that the syntactic tags are helpful to guide the model propagate sentiments of words towards target. Although the dependency results are also not precise enough, the composition selection is automatically learned from data. Hence AdaRNN is more robust for the imprecision of parsing results than the hand-crafted rules. The performances become better after adding the uni-gram and bi-gram features (target-independent).

4.1 Effects of β

We compare different β for AdaRNN defined in Equation (3) in this section. Different parameter β leads to different composition selection schemes.

As illustrated in Figure 3, the AdaRNN-w/oE and AdaRNN-w/E achieve the best accuracies at $\beta = 2$, and they have a similar trend. Specifically, $\beta = 0$ obtains a uniform distribution over the composition functions which does not help improve performances. $\beta \rightarrow \infty$ results in a maximum probability selection algorithm, i.e., only the composition function which has the maximum probability is used. This selection scheme makes

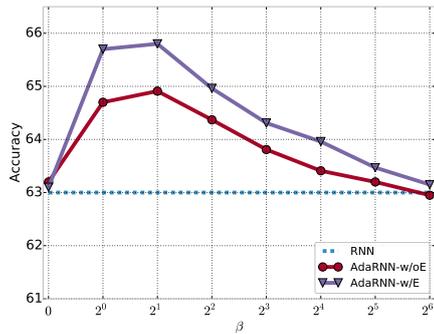


Figure 3: The curve shows the accuracy as the hyper-parameter $\beta = 0, 2^0, 2^1, \dots, 2^6$ increases. AdaRNN achieves the best results at $\beta = 2^1$.

the optimization instable. The performances of $\beta = 1, 2$ are similar and they are better than other settings. It indicates that adaptive selection method is useful to model the compositions. The hyper-parameter β makes trade-offs between uniform selection and maximum selection. It adjusts the effects of these two perspectives.

5 Conclusion

We propose Adaptive Recursive Neural Network (AdaRNN) for the target-dependent Twitter sentiment classification. AdaRNN employs more than one composition functions and adaptively chooses them depending on the context and linguistic tags. For a given tweet, we first convert its dependency tree for the interested target. Next, the AdaRNN learns how to adaptively propagate the sentiments of words to the target node. AdaRNN enables the sentiment propagations to be sensitive to both linguistic and semantic categories by using different compositions. The experimental results illustrate that AdaRNN improves the baselines without hand-crafted rules.

Acknowledgments

This research was partly supported by the National 863 Program of China (No. 2012AA011005), the fund of SKLSDE (Grant No. SKLSDE-2013ZX-06), and Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20111102110019).

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *JMLR*, 3:1137–1155, March.

- Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2014. Adaptive multi-compositionality for recursive neural models with applications to sentiment analysis. In *Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*. AAAI.
- John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12:2121–2159, July.
- Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, June.
- H.O. Georgii. 2011. *Gibbs Measures and Phase Transitions*. De Gruyter studies in mathematics. De Gruyter.
- Kevin Gimpel, Nathan Schneider, Brendan O’Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT ’11, pages 42–47, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Geoffrey E. Hinton. 1986. Learning distributed representations of concepts. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pages 1–12. Hillsdale, NJ: Erlbaum.
- Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’04, pages 168–177, New York, NY, USA. ACM.
- 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: Human Language Technologies - Volume 1*, HLT ’11, pages 151–160, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL ’03, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. 2010. Dependency tree-based sentiment classification using crfs with hidden variables. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 786–794. Association for Computational Linguistics.
- Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics.
- D.E. Rumelhart, G.E. Hinton, and R.J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.
- Richard Socher, Cliff C. Lin, Andrew Y. Ng, and Christopher D. Manning. 2011. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *ICML*.
- Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *EMNLP-CoNLL*, pages 1201–1211.
- Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *EMNLP*, pages 1631–1642.
- Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Comput. Linguist.*, 37(2):267–307, June.

Sprinkling Topics for Weakly Supervised Text Classification

Swapnil Hingmire^{1,2}

swapnil.hingmire@tcs.com

Sutanu Chakraborti²

sutanuc@cse.iitm.ac.in

¹Systems Research Lab, Tata Research Development and Design Center, Pune, India

²Department of Computer Science and Engineering,
Indian Institute of Technology Madras, Chennai, India

Abstract

Supervised text classification algorithms require a large number of documents labeled by humans, that involve a labor-intensive and time consuming process. In this paper, we propose a weakly supervised algorithm in which supervision comes in the form of labeling of Latent Dirichlet Allocation (LDA) topics. We then use this weak supervision to “sprinkle” artificial words to the training documents to identify topics in accordance with the underlying class structure of the corpus based on the higher order word associations. We evaluate this approach to improve performance of text classification on three real world datasets.

1 Introduction

In supervised text classification learning algorithms, the learner (a program) takes human labeled documents as input and learns a decision function that can classify a previously unseen document to one of the predefined classes. Usually a large number of documents labeled by humans are used by the learner to classify unseen documents with adequate accuracy. Unfortunately, labeling a large number of documents is a labor-intensive and time consuming process.

In this paper, we propose a text classification algorithm based on Latent Dirichlet Allocation (LDA) (Blei et al., 2003) which does not need labeled documents. LDA is an unsupervised probabilistic topic model and it is widely used to discover latent semantic structure of a document collection by modeling words in the documents. Blei et al. (Blei et al., 2003) used LDA topics as features in text classification, but they use labeled documents while learning a classifier. sLDA (Blei and McAuliffe, 2007), DiscLDA (Lacoste-Julien

et al., 2008) and MedLDA (Zhu et al., 2009) are few extensions of LDA which model both class labels and words in the documents. These models can be used for text classification, but they need expensive labeled documents.

An approach that is less demanding in terms of knowledge engineering is ClassifyLDA (Hingmire et al., 2013). In this approach, a topic model on a given set of unlabeled training documents is constructed using LDA, then an annotator assigns a class label to some topics based on their most probable words. These labeled topics are used to create a new topic model such that in the new model topics are better aligned to class labels. A class label is assigned to a test document on the basis of its most prominent topics. We extend ClassifyLDA algorithm by “sprinkling” topics to unlabeled documents.

Sprinkling (Chakraborti et al., 2007) integrates class labels of documents into Latent Semantic Indexing (LSI)(Deerwester et al., 1990). The basic idea involves encoding of class labels as artificial words which are “sprinkled” (appended) to training documents. As LSI uses higher order word associations (Kontostathis and Pottenger, 2006), sprinkling of artificial words gives better and class-enriched latent semantic structure. However, Sprinkled LSI is a supervised technique and hence it requires expensive labeled documents. The paper revolves around the idea of labeling topics (which are far fewer in number compared to documents) as in ClassifyLDA, and using these labeled topic for sprinkling.

As in ClassifyLDA, we ask an annotator to assign class labels to a set of topics inferred on the unlabeled training documents. We use the labeled topics to find probability distribution of each training document over the class labels. We create a set of artificial words corresponding to a class label and add (or sprinkle) them to the document. The number of such artificial terms is propor-

tional to the probability of generating the document by the class label. We then infer a set of topics on the sprinkled training documents. As LDA uses higher order word associations (Lee et al., 2010) while discovering topics, we hypothesize that sprinkling will improve text classification performance of ClassifyLDA. We experimentally verify this hypothesis on three real world datasets.

2 Related Work

Several researchers have proposed semi-supervised text classification algorithms with the aim of reducing the time, effort and cost involved in labeling documents. These algorithms can be broadly categorized into three categories depending on how supervision is provided. In the first category, a small set of labeled documents and a large set of unlabeled documents is used while learning a classifier. Semi-supervised text classification algorithms proposed in (Nigam et al., 2000), (Joachims, 1999), (Zhu and Ghahramani, 2002) and (Blum and Mitchell, 1998) are a few examples of this type. However, these algorithms are sensitive to initial labeled documents and hyper-parameters of the algorithm.

In the second category, supervision comes in the form of labeled words (features). (Liu et al., 2004) and (Druck et al., 2008) are a few examples of this type. An important limitation of these algorithms is coming up with a small set of words that should be presented to the annotators for labeling. Also a human annotator may discard or mislabel a polysemous word, which may affect the performance of a text classifier.

The third type of semi-supervised text classification algorithms is based on active learning. In active learning, particular unlabeled documents or features are selected and queried to an oracle (e.g. human annotator). (Godbole et al., 2004), (Raghavan et al., 2006), (Druck et al., 2009) are a few examples of active learning based text classification algorithms. However, these algorithms are sensitive to the sampling strategy used to query documents or features.

In our approach, an annotator does not label documents or words, rather she labels a small set of interpretable topics which are inferred in an unsupervised manner. These topics are very few, when compared to the number of documents. As the most probable words of topics are representative of the dataset, there is no need for the annota-

tor to search for the right set of features for each class. As LDA topics are semantically more meaningful than individual words and can be acquired easily, our approach overcomes limitations of the semi-supervised methods discussed above.

3 Background

3.1 LDA

LDA is an unsupervised probabilistic generative model for collections of discrete data such as text documents. The generative process of LDA can be described as follows:

1. for each topic t , draw a distribution over words: $\phi_t \sim \text{Dirichlet}(\beta_w)$
2. for each document $d \in D$
 - a. Draw a vector of topic proportions: $\theta_d \sim \text{Dirichlet}(\alpha_t)$
 - b. for each word w at position n in d
 - i. Draw a topic assignment: $z_{d,n} \sim \text{Multinomial}(\theta_d)$
 - ii. Draw a word: $w_{d,n} \sim \text{Multinomial}(z_{d,n})$

Where, T is the number of topics, ϕ_t is the word probabilities for topic t , θ_d is the topic probability distribution, $z_{d,n}$ is topic assignment and $w_{d,n}$ is word assignment for n th word position in document d respectively. α_t and β_w are topic and word Dirichlet priors.

The key problem in LDA is posterior inference. The posterior inference involves the inference of the hidden topic structure given the observed documents. However, computing the exact posterior inference is intractable. In this paper we estimate approximate posterior inference using collapsed Gibbs sampling (Griffiths and Steyvers, 2004).

The Gibbs sampling equation used to update the assignment of a topic t to the word $w \in W$ at the position n in document d , conditioned on α_t, β_w is:

$$P(z_{d,n} = t | z_{d,-n}, w_{d,n} = w, \alpha_t, \beta_w) \propto \frac{\psi_{w,t} + \beta_w - 1}{\sum_{v \in W} \psi_{v,t} + \beta_v - 1} \times (\Omega_{t,d} + \alpha_t - 1) \quad (1)$$

where $\psi_{w,c}$ is the count of the word w assigned to the topic c , $\Omega_{c,d}$ is the count of the topic c assigned to words in the document d and W is the vocabulary of the corpus. We use a subscript $d, -n$ to denote the current token, $z_{d,n}$ is ignored in the Gibbs sampling update. After performing collapsed Gibbs sampling using equation 1, we use word topic assignments to compute a point

estimate of the distribution over words $\phi_{w,c}$ and a point estimate of the posterior distribution over topics for each document d (θ_d) is:

$$\phi_{w,t} = \frac{\psi_{w,t} + \beta_w}{\sum_{v \in W} \psi_{v,t} + \beta_v} \quad \theta_{t,d} = \frac{\Omega_{t,d} + \alpha_t}{\sum_{i=1}^T \Omega_{i,d} + \alpha_i} \quad (2) \quad (3)$$

Let $M_D = \langle Z, \Phi, \Theta \rangle$ be the hidden topic structure, where Z is per word per document topic assignment, $\Phi = \{\phi_t\}$ and $\Theta = \{\theta_d\}$.

3.2 Sprinkling

(Chakraborti et al., 2007) propose a simple approach called ‘‘sprinkling’’ to incorporate class labels of documents into LSI. In sprinkling, a set of artificial words are appended to a training document which are specific to the class label of the document. Consider a case of binary classification with classes c_1 and c_2 . If a document d belongs to the class c_1 then a set of artificial words which represent the class c_1 are appended into the document d , otherwise a set of artificial words which represent the class c_2 are appended.

Singular Value Decomposition (SVD) is then performed on the sprinkled training documents and a lower rank approximation is constructed by ignoring dimensions corresponding to lower singular values. Then, the sprinkled terms are removed from the lower rank approximation. (Chakraborti et al., 2007) empirically show that sprinkled words boost higher order word associations and projects documents with same class labels close to each other in latent semantic space.

4 Topic Sprinkling in LDA

In our text classification algorithm, we first infer a set of topics on the given unlabeled document corpus. We then ask a human annotator to assign one or more class labels to the topics based on their most probable words. We use these labeled topics to create a new LDA model as follows. If the topic assigned to the word w at the position n in document d is t , then we replace it by the class label assigned to the topic t . If more than one class labels are assigned to the topic t , then we randomly select one of the class labels assigned to the topic t . If the annotator is unable to label a topic then we randomly select a class label from the set of all class labels. We then update the new LDA model using collapsed Gibbs sampling.

We use this new model to infer the probability distribution of each unlabeled training document over the class labels. Let, $\theta_{c,d}$ be the probability of generating document d by class c . We then sprinkle s artificial words of class label c to document d , such that $s = K * \theta_{c,d}$ for some constant K .

We then infer a set of $|C|$ number of topics on the sprinkled dataset using collapsed Gibbs sampling, where C is the set of class labels of the training documents. We modify collapsed Gibbs sampling update in Equation 1 to carry class label information while inferring topics. If a word in a document is a sprinkled word then while sampling a class label for it, we sample the class label associated with the sprinkled word, otherwise we sample a class label for the word using Gibbs update in Equation 1.

We name this model as Topic Sprinkled LDA (TS-LDA). While classifying a test document, its probability distribution over class labels is inferred using TS-LDA model and it is classified to its most probable class label. Algorithm for TS-LDA is summarized in Table 1.

5 Experimental Evaluation

We determine the effectiveness of our algorithm in relation to ClassifyLDA algorithm proposed in (Hingmire et al., 2013). We evaluate and compare our text classification algorithm by computing Macro averaged F1. As the inference of LDA is approximate, we repeat all the experiments for each dataset ten times and report average Macro-F1. Similar to (Blei et al., 2003) we also learn supervised SVM classifier (LDA-SVM) for each dataset using topics as features and report average Macro-F1.

5.1 Datasets

We use the following datasets in our experiments.

1. **20 Newsgroups:** This dataset contains messages across twenty newsgroups. In our experiments, we use *bydate* version of the 20Newsgroup dataset¹. This version of the dataset is divided into training (60%) and test (40%) datasets. We construct classifiers on training datasets and evaluate them on test datasets.

2. **SRAA: Simulated/Real/Aviation/Auto UseNet data**²: This dataset contains 73,218

¹<http://qwone.com/~jason/20Newsgroups/>

²<http://people.cs.umass.edu/~mccallum/data.html>

-
- **Input:** unlabeled document corpus- D , number of topics- T and number of sprinkled terms- K
1. Infer T number of topics on D for LDA using collapsed Gibbs sampling. Let M_D be the hidden topic structure of this model.
 2. Ask an annotator to assign one or more class labels $c_i \in C$ to a topic based on its 30 most probable words.
 3. **Initialization:** For n th word in document $d \in D$ if $z_{d,n} = t$ and the annotator has labeled topic t with c_i then, $z_{d,n} = c_i$
 4. Update M_D using collapsed Gibbs sampling update in Equation 1.
 5. **Sprinkling:** For each document $d \in D$:
 - (a) Infer a probability distribution θ_d over class labels using M_D using Equation 3.
 - (b) Let, $\theta_{c,d}$ be probability of generating document d by class c .
 - (c) Insert $K * \theta_{c,d}$ distinct words associated with the class c to the document d .
 6. Infer $|C|$ number of topics on the sprinkled document corpus D using collapsed Gibbs sampling update.
 7. Let M'_D be the new hidden topic structure. Let us call this hidden structure as TS-LDA.
 8. **Classification of an unlabeled document d**
 - (a) Infer θ'_d for document d using M'_D .
 - (b) $k = \operatorname{argmax}_i \theta'_{i,d}$
 - (c) $y_d = c_k$
-

Table 1: Algorithm for sprinkling LDA topics for text classification

UseNet articles from four discussion groups, for simulated auto racing (sim_auto), simulated aviation (sim_aviation), real autos (real_auto), real aviation (real_aviation). Following are the three classification tasks associated with this dataset.

1. sim_auto vs sim_aviation vs real_auto vs real_aviation
2. auto (sim_auto + real_auto) vs aviation (sim_aviation + real_aviation)
3. simulated (sim_auto + sim_aviation) vs real (real_auto + real_aviation)

We randomly split SRAA dataset such that 80% is used as training data and remaining is used as test data.

3. **WebKB:** The WebKB dataset³ contains 8145 web pages gathered from university computer

³<http://www.cs.cmu.edu/~webkb/>

science departments. The task is to classify the webpages as *student*, *course*, *faculty* or *project*. We randomly split this dataset such that 80% is used as training and 20% is used as test data.

We preprocess these datasets by removing HTML tags and stop-words.

For various subsets of the 20Newsgroups and WebKB datasets discussed above, we choose number of topics as twice the number of classes. For SRAA dataset we infer 8 topics on the training dataset and label these 8 topics for all the three classification tasks. While labeling a topic, we show its 30 most probable words to the human annotator.

Similar to (Griffiths and Steyvers, 2004), we set symmetric Dirichlet word prior (β_w) for each topic to 0.01 and symmetric Dirichlet topic prior (α_t) for each document to $50/T$, where T is number of topics. We set K i.e. maximum number of words sprinkled per class to 10.

5.2 Results

Table 2 shows experimental results. We can observe that, TS-LDA performs better than ClassifyLDA in 5 of the total 9 subsets. For the *comp-religion-sci* dataset TS-LDA and ClassifyLDA have the same performance. However, ClassifyLDA performs better than TS-LDA for the three classification tasks of SRAA dataset. We can also observe that, performance of TS-LDA is close to supervised LDA-SVM. We should note here that in TS-LDA, the annotator only labels a few topics and not a single document. Hence, our approach exerts a low cognitive load on the annotator, at the same time achieves text classification performance close to LDA-SVM which needs labeled documents.

5.3 Example

Table 3 shows most prominent words of four topics inferred on the *med-space* subset of the 20Newsgroup dataset. We can observe here that most prominent words of the first topic do not represent a single class, while other topics represent either *med* (*medical*) or *space* class. We can say here that, these topics are not ‘‘coherent’’.

We use these labeled topics and create a TS-LDA model using the algorithm described in Table 1. Table 4 shows words corresponding to the top two topics of the TS-LDA model. We can observe here that these two topics are more coherent than the topics in Table 3.

Dataset	# Topics	Text Classification (Macro-F1)		
		ClassifyLDA	TS-LDA	LDA-SVM
20Newsgroups				
med-space	4	0.892	0.938	0.933
politics-religion	4	0.836	0.897	0.901
politics-sci	4	0.887	0.901	0.910
comp-religion-sci	6	0.853	0.853	0.872
politics-rec-religion-sci	8	0.842	0.858	0.862
SRAA				
real_auto-real_aviation-sim_auto-sim_aviation	8	0.766	0.741	0.820
auto-aviation	8	0.926	0.910	0.934
real-sim	8	0.918	0.902	0.923
WebKB				
WebKB	8	0.627	0.672	0.730

Table 2: Experimental results of text classification on various datasets.

ID	Most prominent words in the topic	Class (med / space)
0	science scientific idea large theory bit pat thought problem isn	med + space
1	information health research medical water cancer hiv aids children institute newsletter	med
2	msg food doctor disease pain day treatment blood steve dyer medicine symptoms	med
3	space nasa launch earth orbit moon shuttle data lunar satellite	space

Table 3: Topic labeling on the *med-space* subset of the 20Newsgroup dataset

ID	Most prominent words in the topic	Class (med / space)
0	msg medical health food disease years problem information doctor pain cancer	med
1	space launch earth data orbit moon program shuttle lunar satellite	space

Table 4: Topics inferred on the *med-space* subset of the 20Newsgroup dataset after sprinkling labeled topics from Table 3.

Hence, we can say here that, in addition to text classification, sprinkling improves coherence of topics.

We should note here that, in ClassifyLDA, the annotator is able to assign a single class label to a topic. If the annotator assigns a wrong class label to a topic representing multiple classes (e.g. first topic in Table 3), then it may affect the performance of the resulting classifier. However, in our approach the annotator can assign multiple class labels to a topic, hence our approach is more flexible for the annotator to encode her domain knowledge efficiently.

6 Conclusions and Future Work

In this paper we propose a novel algorithm that classifies documents based on class labels over few topics. This reduces the need to label a large collection of documents. We have used the idea of sprinkling originally proposed in the context of supervised Latent Semantic Analysis, but the setting here is quite different. Unlike the work in (Chakraborti et al., 2007), we do not assume that we have class labels over the set of training documents. Instead, to realize our goal of reducing knowledge acquisition overhead, we propose a way of propagating knowledge of few topic labels to the words and inducing a new topic distribution that has its topics more closely aligned to the class labels. The results show that the approach can yield performance comparable to entirely supervised settings. In future work, we also envision the possibility of sprinkling knowledge from background knowledge sources like Wikipedia (Gabrilovich and Markovitch, 2007) to realize an alignment of topics to Wikipedia concepts. We would like to study effect of change in number of topics on the text classification performance. We will also explore techniques which will help annotators to encode their domain knowledge efficiently when the topics are not well aligned to the class labels.

References

- David M. Blei and Jon D. McAuliffe. 2007. Supervised Topic Models. In *NIPS*.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, 3:993–1022, March.

- Avrim Blum and Tom Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-Training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100.
- Sutanu Chakraborti, Rahman Mukras, Robert Lothian, Nirmalie Wiratunga, Stuart N. K. Watt, David J. Harper. 2007. Supervised Latent Semantic Indexing Using Adaptive Sprinkling. In *IJCAI*, pages 1582–1587.
- Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *JASIS*, 41(6):391–407.
- Gregory Druck, Gideon Mann, and Andrew McCallum. 2008. Learning from Labeled Features using Generalized Expectation criteria. In *SIGIR*, pages 595–602.
- Gregory Druck, Burr Settles, and Andrew McCallum. 2009. Active Learning by Labeling Features. In *EMNLP*, pages 81–90.
- Shantanu Godbole, Abhay Harpale, Sunita Sarawagi, and Soumen Chakrabarti. 2004. Document Classification through Interactive Supervision of Document and Term Labels. In *PKDD*, pages 185–196.
- Thomas L. Griffiths and Mark Steyvers. 2004. Finding Scientific Topics. *PNAS*, 101(suppl. 1):5228–5235, April.
- Swapnil Hingmire, Sandeep Chougule, Girish K. Palshikar, and Sutanu Chakraborti. 2013. Document Classification by Topic Labeling. In *SIGIR*, pages 877–880.
- Thorsten Joachims. 1999. Transductive Inference for Text Classification using Support Vector Machines. In *ICML*, pages 200–209.
- April Kontostathis and William M. Pottenger. 2006. A Framework for Understanding Latent Semantic Indexing (LSI) Performance. *Inf. Process. Manage.*, 42(1):56–73, January.
- Simon Lacoste-Julien, Fei Sha, and Michael I. Jordan. 2008. DiscLDA: Discriminative Learning for Dimensionality Reduction and Classification. In *NIPS*.
- Sangno Lee, Jeff Baker, Jaeki Song, and James C. Wetherbe. 2010. An Empirical Comparison of Four Text Mining Methods. In *Proceedings of the 2010 43rd Hawaii International Conference on System Sciences*, pages 1–10.
- Bing Liu, Xiaoli Li, Wee Sun Lee, and Philip S. Yu. 2004. Text Classification by Labeling Words. In *Proceedings of the 19th national conference on Artificial intelligence*, pages 425–430.
- Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning - Special issue on information retrieval*, 39(2-3), May-June.
- Hema Raghavan, Omid Madani, and Rosie Jones. 2006. Active Learning with Feedback on Features and Instances. *JMLR*, 7:1655–1686, December.
- Xiaojin Zhu and Zoubin Ghahramani. 2002. Learning from Labeled and Unlabeled Data with Label Propagation. Technical report, Carnegie Mellon University.
- Jun Zhu, Amr Ahmed, and Eric P. Xing. 2009. MedLDA: Maximum Margin Supervised Topic Models for Regression and Classification. In *ICML*, pages 1257–1264.
- Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *IJCAI*, pages 1606–1611.

A Feature-Enriched Tree Kernel for Relation Extraction

Le Sun and Xianpei Han

State Key Laboratory of Computer Science
Institute of Software, Chinese Academy of Sciences
HaiDian District, Beijing, China.

{sunle, xianpei}@nfs.iscas.ac.cn

Abstract

Tree kernel is an effective technique for relation extraction. However, the traditional syntactic tree representation is often *too coarse* or *ambiguous* to accurately capture the semantic relation information between two entities. In this paper, we propose a new tree kernel, called *feature-enriched tree kernel (FTK)*, which can enhance the traditional tree kernel by: 1) refining the syntactic tree representation by annotating each tree node with a set of discriminant features; and 2) proposing a new tree kernel which can better measure the syntactic tree similarity by taking all features into consideration. Experimental results show that our method can achieve a 5.4% F-measure improvement over the traditional convolution tree kernel.

1 Introduction

Relation Extraction (RE) aims to identify a set of predefined relations between pairs of entities in text. In recent years, relation extraction has received considerable research attention. An effective technique is the tree kernel (Zelenko et al., 2003; Zhou et al., 2007; Zhang et al., 2006; Qian et al., 2008), which can exploit syntactic parse tree information for relation extraction. Given a pair of entities in a sentence, the tree kernel-based RE method first represents the relation information between them using a proper sub-tree (e.g., SPT – the sub-tree enclosed by the shortest path linking the two involved entities). For example, the three syntactic tree representations in Figure 1. Then the similarity between two trees are computed using a tree kernel, e.g., the convolution tree kernel proposed by Collins and Duffy (2001). Finally, new relation instances are extracted using kernel based classifiers, e.g., the SVM classifier.

Unfortunately, one *main* shortcoming of the traditional tree kernel is that the syntactic tree representation usually cannot accurately capture the

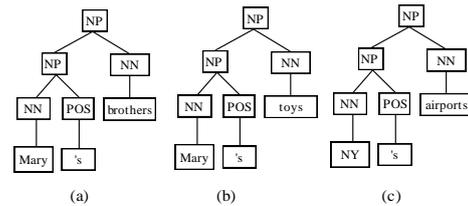


Figure 1. The ambiguity of possessive structure

relation information between two entities. This is mainly due to the following two reasons:

1) *The syntactic tree focuses on representing syntactic relation/structure, which is often too coarse or ambiguous to capture the semantic relation information.* In a syntactic tree, each node indicates a clause/phrase/word and is only labeled with a Treebank tag (Marcus et al., 1993). The Treebank tag, unfortunately, is usually too coarse or too general to capture semantic information. For example, all the three trees in Figure 1 share the same possessive syntactic structure, but express quite different semantic relations: where “*Mary’s brothers*” expresses *PER-SOC Family* relation, “*Mary’s toys*” expresses *Possession* relation, and “*New York’s airports*” expresses *PHYS-Located* relation.

2) *Some critical information may be lost during sub-tree representation extraction.* For example, in Figure 2, when extracting SPT representation, all nodes outside the shortest-path will be pruned, such as the nodes *[NN plants]* and *[POS ’s]* in tree *T1*. In this pruning process, the critical information “word *town* is the possessor of the possessive phrase *the town’s plants*” will be lost, which in turn will lead to the misclassification of the *DISC* relation between *one* and *town*.

This paper proposes a new tree kernel, referred to as *feature-enriched tree kernel (FTK)*, which can effectively resolve the above problems by enhancing the traditional tree kernel in the following ways:

1) We refine the syntactic tree representation by annotating each tree node with a set of discriminant features. These features are utilized to

better capture the semantic relation information between two entities. For example, in order to differentiate the syntactic tree representations in Figure 1, FTK will annotate them with several features indicating “brother is a male sibling”, “toy is an artifact”, “New York is a city”, “airport is facility”, etc.

2) Based on the refined syntactic tree representation, we propose a new tree kernel – *feature-enriched tree kernel*, which can better measure the similarity between two trees by also taking all features into consideration.

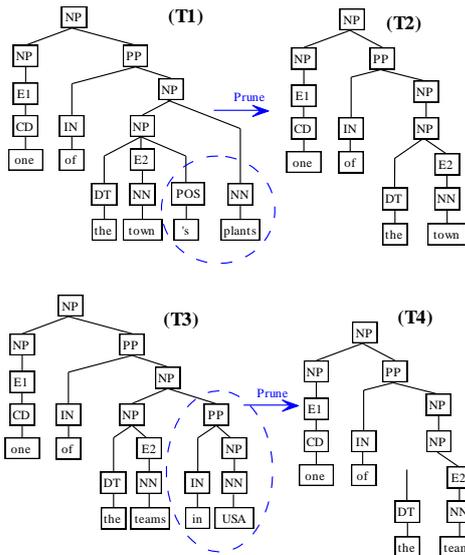


Figure 2. SPT representation extraction

We have experimented our method on the ACE 2004 RDC corpus. Experimental results show that our method can achieve a 5.4% F-measure improvement over the traditional convolution tree kernel based method.

This paper is organized as follows. Section 2 describes the feature-enriched tree kernel. Section 3 presents the features we used. Section 4 discusses the experiments. Section 5 briefly reviews the related work. Finally Section 6 concludes this paper.

2 The Feature-Enriched Tree Kernel

In this section, we describe the proposed feature-enriched tree kernel (FTK) for relation extraction.

2.1 Refining Syntactic Tree Representation

As described in above, syntactic tree is often too coarse or too ambiguous to represent the semantic relation information between two entities. To resolve this problem, we refine the syntactic tree representation by annotating each tree node with a set of discriminant features.

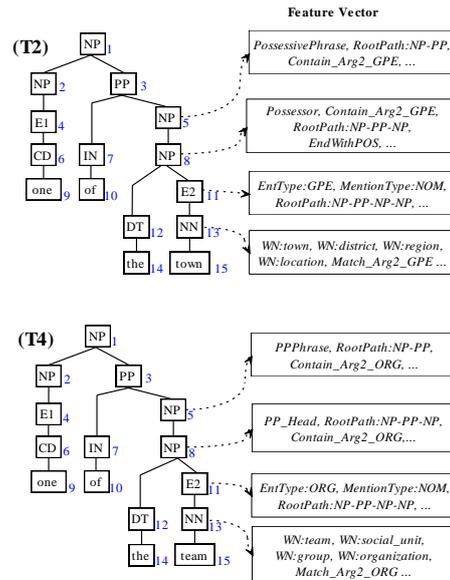


Figure 3. Syntactic tree enriched with features

Specifically, for each node n in a syntactic tree T , we represent it as a tuple:

$$R_n = (L_n, F_n)$$

where L_n is its phrase label (i.e., its Treebank tag), and F_n is a feature vector which indicates the characteristics of node n , which is represented as:

$$F_n = \{f_1, f_2, \dots, f_N\}$$

where f_i is a feature and is associated with a weight $w_i \in (0, 1)$. The feature we used includes characteristics of relation instance, phrase properties and context information (See Section 3 for details).

For demonstration, Figure 3 shows the feature-enriched version of tree T_2 and tree T_4 in Figure 2. We can see that, although T_2 and T_4 share the same syntactic structure, the annotated features can still differentiate them. For example, the NP_5 node in tree T_2 and the NP_5 node in tree T_4 are differentiated using their features *PossessivePhrase* and *PPPhrase*, which indicate that NP_5 in T_2 is a possessive phrase, meanwhile NP_5 in T_4 is a preposition phrase.

2.2 Feature-Enriched Tree Kernel

This section describes how to take into account the annotated features for a better tree similarity.

In Collins and Duffy’s convolution tree kernel (CTK), the similarity between two trees T_1 and T_2 is the number of their common sub-trees:

$$K_C(T_1, T_2) = \sum_{t_1 \in T_1} \sum_{t_2 \in T_2} \delta(t_1, t_2)$$

Using this formula, CTK only considers whether two enumerated sub-trees have the identical syntactic structure (the indicator $\delta(t_1, t_2)$ is 1 if the

two sub-trees t_1 and t_2 have the identical syntactic structure and 0 otherwise). Such an assumption makes CTK can only capture the syntactic structure similarity between two trees, while ignoring other useful information.

To resolve the above problem, the *feature-enriched tree kernel (FTK)* compute the similarity between two trees as the sum of the similarities between their common sub-trees:

$$F_{tk}(T_1, T_2) = \sum_{t_1 \in T_1} \sum_{t_2 \in T_2} k(t_1, t_2)$$

where $k(t_1, t_2)$ is the similarity between enumerated sub-trees t_1 and t_2 , which is computed as:

$$k(t_1, t_2) = \delta(t_1, t_2) \times \prod_{(n_i, n_j) \in E(t_1, t_2)} (1 + \text{sim}(n_i, n_j))$$

where $\delta(t_1, t_2)$ is the same indicator function as in CTK; (n_i, n_j) is a pair of aligned nodes between t_1 and t_2 , where n_i and n_j are correspondingly in the same position of tree t_1 and t_2 ; $E(t_1, t_2)$ is the set of all aligned node pairs; $\text{sim}(n_i, n_j)$ is the feature vector similarity between node n_i and n_j , computed as the dot product between their feature vectors F_{n_i} and F_{n_j} .

Notice that, if all nodes are not annotated with features, $k(t_1, t_2)$ will be equal to $\delta(t_1, t_2)$. In this perspective, we can view $k(t_1, t_2)$ as a similarity adjusted version of $\delta(t_1, t_2)$, i.e., $\delta(t_1, t_2)$ only considers whether two nodes are equal, in contrast $k(t_1, t_2)$ further considers the feature similarity $\text{sim}(n_i, n_j)$ between two nodes.

The Computation of FTK. As the same as CTK, FTK can be efficiently computed as:

$$F_{tk}(T_1, T_2) = \sum_{n_1 \in N_1, n_2 \in N_2} \Delta(n_1, n_2)$$

where N_j is the set of nodes in tree T_j , and $\Delta(n_1, n_2)$ evaluates the sum of the similarities of common sub-trees rooted at node n_1 and node n_2 , which is recursively computed as follows:

- 1) If the production rules of n_1 and n_2 are different, $\Delta(n_1, n_2) = 0$;
- 2) If both n_1 and n_2 is pre-terminal nodes, $\Delta(n_1, n_2) = (1 + \text{sim}(n_1, n_2)) \times \lambda$;
Otherwise go to step 3;
- 3) Calculate $\Delta(n_1, n_2)$ recursively as:

$$\Delta(n_1, n_2) = \lambda \times (1 + \text{sim}(n_1, n_2)) \times \sum_{k=1}^{\#ch(n_1)} (1 + \Delta(ch(n_1, k), ch(n_2, k)))$$

3 Features for Relation Extraction

This section presents the features we used to enrich the syntactic tree representation.

3.1 Instance Feature

Relation instances of the same type often share some common characteristics. In this paper, we add the following instance features to the root node of a sub-tree representation:

1) **Syntactico-Semantic structure.** A feature indicates whether a relation instance has the following four syntactico-semantic structures in (Chan & Roth, 2011) – *Premodifiers*, *Possessive*, *Preposition*, *Formulaic* and *Verbal*.

2) **Entity-related information of arguments.** Features about the entity information of arguments, including: a) *#TP1-#TP2*: the concat of the major entity types of arguments; b) *#ST1-#ST2*: the concat of the sub entity types of arguments; c) *#MT1-#MT2*: the concat of the mention types of arguments.

3) **Base phrase chunking features.** Features about the phrase path between two arguments and the phrases' head before and after the arguments, which are the same as the phrase chunking features in (Zhou, et al., 2005).

3.2 Phrase Feature

As discussed in above, the Treebank tag is too coarse to capture the property of a phrase node. Therefore, we enrich each phrase node with features about its lexical pattern, its content information, and its lexical semantics:

1) **Lexical Pattern.** We capture the lexical pattern of a phrase node using the following features: a) *LP_Poss*: A feature indicates the node is a possessive phrase; b) *LP_PP*: A feature indicates the node is a preposition phrase; c) *LP_CC*: A feature indicates the node is a conjunction phrase; d) *LP_EndWithPUNC*: A feature indicates the node ends with a punctuation; e) *LP_EndWithPOSS*: A feature indicates the node ends with a possessive word.

2) **Content Information.** We capture the property of a node's content using the following features: a) *MB_#Num*: The number of mentions contained in the phrase; b) *MB_C_#Type*: A feature indicates that the phrase contains a mention with major entity type *#Type*; c) *MW_#Num*: The number of words within the phrase.

3) **Lexical Semantics.** If the node is a pre-terminal node, we capture its lexical semantic by adding features indicating its WordNet sense information. Specifically, the first WordNet sense of the terminal word, and all this sense's hyponym senses will be added as features. For example, WordNet senses *{New York#1, city#1, district#1,*

region#1, ... } will be added as features to the *[NN New York]* node in Figure 1.

3.3 Context Information Feature

The context information of a phrase node is critical for identifying the role and the importance of a sub-tree in the whole relation instance. This paper captures the following context information:

1) **Contextual path from sub-tree root to the phrase node.** As shown in Zhou et al. (2007), the context path from root to the phrase node is an effective context information feature. In this paper, we use the same settings in (Zhou et al., 2007), i.e., each phrase node is enriched with its context paths of length 1, 2, 3.

2) **Relative position with arguments.** We observed that a phrase’s relative position with the relation’s arguments is useful for identifying the role of the phrase node in the whole relation instance. To capture the relative position information, we define five possible relative positions between a phrase node and an argument, corresponding *match*, *cover*, *within*, *overlap* and *other*. Using these five relative positions, we capture the context information using the following features:

a) *#RP_Arg1Head_#Arg1Type*: a feature indicates the relative position of a phrase node with argument 1’s head phrase, where *#RP* is the relative position (one of *match*, *cover*, *within*, *overlap*, *other*), and *#Arg1Type* is the major entity type of argument 1. One example feature may be *Match_Arg1Head_LOC*.

b) *#RP_Arg2Head_#Arg2Type*: The relative position with argument 2’s head phrase;

c) *#RP_Arg1Extend_#Arg1Type*: The relative position with argument 1’s extended phrase;

d) *#PR_Arg2Extend_#Arg2Type*: The relative position with argument 2’s extended phrase.

Feature weighting. Currently, we set all features with an uniform weight $w \in (0, 1)$, which is used to control the relative importance of the feature in the final tree similarity: the larger the feature weight, the more important the feature in the final tree similarity.

4 Experiments

4.1 Experimental Setting

To assess the feature-enriched tree kernel, we evaluate our method on the ACE RDC 2004 corpus using the same experimental settings as (Qian et al., 2008). That is, we parse all sentences using the Charniak’s parser (Charniak, 2001), relation instances are generated by iterating over all pairs of entity mentions occurring in the same sentence.

In our experiments, we implement the feature-enriched tree kernel by extending the SVM^{light} (Joachims, 1998) with the proposed tree kernel function (Moschitti, 2004). We apply the *one vs. others* strategy for multiple classification using SVM. For SVM training, the parameter C is set to 2.4 for all experiments, and the tree kernel parameter λ is tuned to 0.2 for FTK and 0.4 (the optimal parameter setting used in Qian et al.(2008)) for CTK.

4.2 Experimental Results

4.2.1 Overall performance

We compare our method with the standard convolution tree kernel (CTK) on the state-of-the-art context sensitive shortest path-enclosed tree representation (CSPT, Zhou et al., 2007). We experiment our method with four different feature settings, correspondingly: 1) FTK with only instance features – *FTK(instance)*; 2) FTK with only phrase features – *FTK(phrase)*; 3) FTK with only context information features – *FTK(context)*; and 4) FTK with all features – *FTK*. The overall performance of CTK and FTK is shown in Table 1, the F-measure improvements over CTK are also shown inside the parentheses. The detailed performance of FTK on the 7 major relation types of ACE 2004 is shown in Table 2.

	P(%)	R(%)	F
CTK	77.1	61.3	68.3 (-----)
FTK(instance)	78.5	64.6	70.9 (+2.6%)
FTK(phrase)	78.3	64.2	70.5 (+2.2%)
FTK(context)	80.1	67.5	73.2 (+4.9%)
FTK	81.2	67.4	73.7 (+5.4%)

Table 1. Overall Performance

Relation Type	P(%)	R(%)	F	Impr
EMP-ORG	84.7	82.4	83.5	5.8%
PER-SOC	79.9	70.7	75.0	1.0%
PHYS	73.3	64.4	68.6	7.0%
ART	83.6	57.5	68.2	1.7%
GPE-AFF	74.7	56.6	64.4	4.3%
DISC	81.6	48.0	60.5	6.6%
OTHER-AFF	74.2	36.8	49.2	1.0%

Table 2. FTK on the 7 major relation types and their F-measure improvement over CTK

From Table 1 and 2, we can see that:

1) By refining the syntactic tree with discriminant features and incorporating these features into the final tree similarity, FTK can significantly improve the relation extraction performance: compared with the convolution tree kernel baseline *CTK*, our method can achieve a 5.4% F-measure improvement.

2) All types of features can improve the performance of relation extraction: FTK can correspondingly get 2.6%, 2.2% and 4.9% F-measure improvements using instance features, phrase features and context information features.

3) Within the three types of features, context information feature can achieve the highest F-measure improvement. We believe this may be because: ① The context information is useful in providing clues for identifying the role and the importance of a sub-tree; and ② The context-free assumption of CTK is too strong, some critical information will be lost in the CTK computation.

4) The performance improvement of FTK varies significantly on different relation types: in Table 2, most performance improvement gains from the *EMP-ORG*, *PHYS*, *GPE-AFF* and *DISC* relation types. We believe this may be because the discriminant features will better complement the syntactic tree for capturing *EMP-ORG*, *PHYS*, *GPE-AFF* and *DISC* relation. On contrast the features may be redundant to the syntactic information for other relation types.

System	P(%)	R(%)	F
Qian et al., (2008): composite kernel	83.0	72.0	77.1
Zhou et al., (2007): composite kernel	82.2	70.2	75.8
Ours: FTK with CSPT	81.2	67.4	73.7
Zhou et al., (2007): context sensitive CTK with CSPT	81.1	66.7	73.2
Ours: FTK with SPT	81.1	66.2	72.9
Jiang & Zhai (2007): MaxEnt classifier with features	74.6	71.3	72.9
Zhang et al., (2006): composite kernel	76.1	68.4	72.1
Zhao & Grishman, (2005): Composite kernel	69.2	70.5	70.4
Zhang et al., (2006): CTK with SPT	74.1	62.4	67.7

Table 3. Comparison of different systems on the ACE RDC 2004 corpus

4.2.2 Comparison with other systems

Finally, Table 3 compares the performance of our method with several other systems. From Table 3, we can see that FTK can achieve competitive performance: ① It achieves a 0.8% F-measure improvement over the feature-based system of Jiang & Zhai (2007); ② It achieves a 0.5% F-measure improvement over a state-of-the-art tree kernel: context sensitive CTK with CSPT of Zhou et al., (2007); ③ The F-measure of our system is slightly lower than the current best performance on ACE 2004 (Qian et al., 2008) – 73.7 vs. 77.1, we believe this is because the system of (Qian et al., 2008) adopts two extra techniques: composing tree kernel with a state-of-the-art feature-based kernel and

using a more proper sub-tree representation. We believe these two techniques can also be used to further improve the performance of our system.

5 Related Work

This section briefly reviews the related work. A classical technique for relation extraction is to model the task as a feature-based classification problem (Kambhatla, 2004; Zhou et al., 2005; Jiang & Zhai, 2007; Chan & Roth, 2010; Chan & Roth, 2011), and feature engineering is obviously the key for performance improvement. As an alternative, tree kernel-based method implicitly defines features by directly measuring the similarity between two structures (Bunescu and Mooney, 2005; Bunescu and Mooney, 2006; Zelenko et al., 2003; Culotta and Sorensen, 2004; Zhang et al., 2006). Composite kernels were also used (Zhao and Grishman, 2005; Zhang et al., 2006).

The main drawback of the current tree kernel is that the syntactic tree representation often cannot accurately capture the relation information. To resolve this problem, Zhou et al. (2007) took the ancestral information of sub-trees into consideration; Reichartz and Korte (2010) incorporated dependency type information into a tree kernel; Plank and Moschitti (2013) and Liu et al. (2013) embedded semantic information into tree kernel. Bloehdorn and Moschitti (2007a, 2007b) proposed Syntactic Semantic Tree Kernels (SSTK), which can capture the semantic similarity between leaf nodes. Moschitti (2009) proposed a tree kernel which specifies a kernel function over any pair of nodes between two trees, and it was further extended and applied in other tasks in (Croce et al., 2011; Croce et al., 2012; Mehdad et al., 2010).

6 Conclusions and Future Work

This paper proposes a *feature-enriched tree kernel*, which can: 1) refine the syntactic tree representation; and 2) better measure the similarity between two trees. For future work, we want to develop a feature weighting algorithm which can accurately measure the relevance of a feature to a relation instance for better RE performance.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants no. 61100152 and 61272324, and the Open Project of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research under Grants no. ICDD201204.

References

- Agichtein, E. and Gravano, L. 2000. *Snowball: Extracting relations from large plain-text collections*. In: Proceedings of the 5th ACM Conference on Digital Libraries, pp. 85–94.
- Plank, B. and Moschitti, A. 2013. *Embedding Semantic Similarity in Tree Kernels for Domain Adaptation of Relation Extraction*. In: Proceedings of ACL 2013.
- Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M. and Etzioni, O. 2007. *Open information extraction from the Web*. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 2670–2676.
- Bunescu, R. and Mooney, R. 2005. *A shortest path dependency kernel for relation extraction*. In: Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing, pp.724–731.
- Bloehdorn, S. and Moschitti, A. 2007a. *Combined Syntactic and Semantic Kernels for Text Classification*. In: Proceedings of the 29th European Conference on Information Retrieval (ECIR).
- Bloehdorn, S. and Moschitti, A. 2007b. *Structure and semantics for expressive text kernels*. In: Proceeding of ACM 16th Conference on Information and Knowledge Management (CIKM).
- Bunescu, R. and Mooney, R., 2006. *Subsequence kernels for relation extraction*. In: Advances in Neural Information Processing Systems 18, pp. 171–178.
- Charniak, E., 2001. *Immediate-head parsing for language models*. In: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics, pp. 124-131.
- Chan, Y. S. and Roth, D. 2010. *Exploiting background knowledge for relation extraction*. In: Proceedings of the 23rd International Conference on Computational Linguistics, pp. 152–160.
- Chan, Y. S. and Roth, D. 2011. *Exploiting syntactico-semantic structures for relation extraction*. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pp. 551–560.
- Croce, D., Moschitti, A. and Basili, R. 2011. *Structured lexical similarity via convolution kernels on dependency trees*. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 1034–1046.
- Croce, D., Moschitti, A., Basili, R. and Palmer, M. 2012. *Verb Classification using Distributional Similarity in Syntactic and Semantic Structures*. In: Proceedings of ACL 2012, pp. 263-272.
- Culotta, A. and Sorensen, J. 2004. *Dependency tree kernels for relation extraction*. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, pp. 423–429.
- Grishman, R. and Sundheim, B. 1996. *Message understanding conference-6: A brief history*. In: Proceedings of the 16th International Conference on Computational Linguistics, pp. 466–471.
- Collins, M. and Duffy, N., 2001. *Convolution Kernels for Natural Language*. In: Proceedings of NIPS 2001.
- Liu, D., et al. 2013. *Incorporating lexical semantic similarity to tree kernel-based Chinese relation extraction*. In: Proceedings of Chinese Lexical Semantics 2013.
- Jiang, J. and Zhai, C. 2007. *A systematic exploration of the feature space for relation extraction*. In: Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, pp. 113–120.
- Joachims, T. 1998. *Text Categorization with Support Vector Machine: learning with many relevant features*. ECML-1998: 137-142.
- Kambhatla, N. 2004. *Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations*. In: the Proceedings of 42st Annual Meeting of the Association for Computational Linguistics, pp. 178–181.
- Krause, S., Li, H., Uszkoreit, H., & Xu, F. 2012. *Large-scale learning of relation-extraction rules with distant supervision from the web*. In: Proceedings of ISWC 2012, pp. 263-278.
- Marcus, M. P., Marcinkiewicz, M. A., & Santorini, B. 1993. *Building a large annotated corpus of English: The Penn Treebank*. Computational linguistics, 19(2), 313-330.
- Moschitti, A. 2004. *A study on Convolution Kernels for Shallow Semantic Parsing*. In: Proceedings of the 42-th Conference on Association for Computational Linguistic (ACL-2004).
- Moschitti, A. 2009. *Syntactic and semantic kernels for short text pair categorization*. In: Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009), pp. 576–584.
- Mehdad, Y., Moschitti, A. and Zanzotto, F. 2010. *Syntactic/Semantic Structures for Textual Entailment Recognition*. In: Proceedings of Human Language Technology - North American chapter of the Association for Computational Linguistics.
- Mintz, M., Bills, S., Snow, R. and Jurafsky D. 2009. *Distant supervision for relation extraction without labeled data*. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pp. 1003–1011.

- Qian L., Zhou G., Kong F., Zhu Q., and Qian P., 2008. *Exploiting constituent dependencies for tree kernel based semantic relation extraction*. In: Proceedings of the 22nd International Conference on Computational Linguistics, pp. 697-704.
- Reichartz, F. and H. Korte, et al. 2010. *Semantic relation extraction with kernels over typed dependency trees*. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining.
- Zelenko, D., Aone, C., and Richardella, A. 2003. *Kernel methods for relation extraction*. Journal of Machine Learning Research, 3:1083–1106.
- Zhang, M., Zhang, J., and Su, J. 2006. *Exploring syntactic features for relation extraction using a convolution tree kernel*. In: Proceedings of the Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics, pages 288–295.
- Zhang, M., Zhang, J., Su, J. and Zhou, G. 2006. *A composite kernel to extract relations between entities with both flat and structured features*. In Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, pages 825–832.
- Zhao, S. and Grishman, R. 2005. *Extracting relations with integrated information using kernel methods*. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, pages 419–426.
- Zhou, G., Su, J., Zhang, J., and Zhang, M. 2005. *Exploring various knowledge in relation extraction*. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, pages 427–434.
- Zhou, G. and Zhang M. 2007. *Extracting relation information from text documents by exploring various types of knowledge*. Information Processing & Management 43(4): 969--982.
- Zhou, G., et al. 2007. *Tree kernel-based relation extraction with context-sensitive structured parse tree information*. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 728–736.

Employing Word Representations and Regularization for Domain Adaptation of Relation Extraction

Thien Huu Nguyen

Computer Science Department
New York University
New York, NY 10003 USA
thien@cs.nyu.edu

Ralph Grishman

Computer Science Department
New York University
New York, NY 10003 USA
grishman@cs.nyu.edu

Abstract

Relation extraction suffers from a performance loss when a model is applied to out-of-domain data. This has fostered the development of domain adaptation techniques for relation extraction. This paper evaluates word embeddings and clustering on adapting feature-based relation extraction systems. We systematically explore various ways to apply word embeddings and show the best adaptation improvement by combining word cluster and word embedding information. Finally, we demonstrate the effectiveness of regularization for the adaptability of relation extractors.

1 Introduction

The goal of Relation Extraction (RE) is to detect and classify relation mentions between entity pairs into predefined relation types such as *Employment* or *Citizenship* relationships. Recent research in this area, whether feature-based (Kambhatla, 2004; Boschee et al., 2005; Zhou et al., 2005; Grishman et al., 2005; Jiang and Zhai, 2007a; Chan and Roth, 2010; Sun et al., 2011) or kernel-based (Zelenko et al., 2003; Bunescu and Mooney, 2005a; Bunescu and Mooney, 2005b; Zhang et al., 2006; Qian et al., 2008; Nguyen et al., 2009), attempts to improve the RE performance by enriching the feature sets from multiple sentence analyses and knowledge resources. The fundamental assumption of these supervised systems is that the training data and the data to which the systems are applied are sampled independently and identically from the same distribution. When there is a mismatch between data distributions, the RE performance of these systems tends to degrade dramatically (Plank and Moschitti, 2013). This is where we need to resort to domain adaptation techniques (DA) to adapt a model trained on one domain (the

source domain) into a new model which can perform well on new domains (the target domains).

The consequences of linguistic variation between training and testing data on NLP tools have been studied extensively in the last couple of years for various NLP tasks such as Part-of-Speech tagging (Blitzer et al., 2006; Huang and Yates, 2010; Schnabel and Schütze, 2014), named entity recognition (Daumé III, 2007) and sentiment analysis (Blitzer et al., 2007; Daumé III, 2007; Daumé III et al., 2010; Blitzer et al., 2011), etc. Unfortunately, there is very little work on domain adaptation for RE. The only study explicitly targeting this problem so far is by Plank and Moschitti (2013) who find that the out-of-domain performance of kernel-based relation extractors can be improved by embedding semantic similarity information generated from word clustering and latent semantic analysis (LSA) into syntactic tree kernels. Although this idea is interesting, it suffers from two major limitations:

- + It does not incorporate word cluster information at different levels of granularity. In fact, Plank and Moschitti (2013) only use the 10-bit cluster prefix in their study. We will demonstrate later that the adaptability of relation extractors can benefit significantly from the addition of word cluster features at various granularities.

- + It is unclear if this approach can encode real-valued features of words (such as word embeddings (Mnih and Hinton, 2007; Collobert and Weston, 2008)) effectively. As the real-valued features are able to capture latent yet useful properties of words, the augmentation of lexical terms with these features is desirable to provide a more general representation, potentially helping relation extractors perform more robustly across domains.

In this work, we propose to avoid these limitations by applying a feature-based approach for RE which allows us to integrate various word features of generalization into a single system more natu-

rally and effectively.

The application of word representations such as word clusters in domain adaptation of RE (Plank and Moschitti, 2013) is motivated by its successes in semi-supervised methods (Chan and Roth, 2010; Sun et al., 2011) where word representations help to reduce data-sparseness of lexical information in the training data. In DA terms, since the vocabularies of the source and target domains are usually different, word representations would mitigate the lexical sparsity by providing general features of words that are shared across domains, hence bridge the gap between domains. The underlying hypothesis here is that the absence of lexical target-domain features in the source domain can be compensated by these general features to improve RE performance on the target domains.

We extend this motivation by further evaluating word embeddings (Bengio et al., 2001; Bengio et al., 2003; Mnih and Hinton, 2007; Collobert and Weston, 2008; Turian et al., 2010) on feature-based methods to adapt RE systems to new domains. We explore the embedding-based features in a principled way and demonstrate that word embedding itself is also an effective representation for domain adaptation of RE. More importantly, we show empirically that word embeddings and word clusters capture different information and their combination would further improve the adaptability of relation extractors.

2 Regularization

Given the more general representations provided by word representations above, how can we learn a relation extractor from the labeled source domain data that generalizes well to new domains? In traditional machine learning where the challenge is to utilize the training data to make predictions on unseen data points (generated from the same distribution as the training data), the classifier with a good generalization performance is the one that not only fits the training data, but also avoids overfitting over it. This is often obtained via regularization methods to penalize complexity of classifiers. Exploiting the shared interest in generalization performance with traditional machine learning, in domain adaptation for RE, we would prefer the relation extractor that fits the source domain data, but also circumvents the overfitting problem

over this source domain¹ so that it could generalize well on new domains. Eventually, regularization methods can be considered naturally as a simple yet general technique to cope with DA problems.

Following Plank and Moschitti (2013), we assume that we only have labeled data in a single source domain but no labeled as well as unlabeled target data. Moreover, we consider the single-system DA setting where we construct a single system able to work robustly with different but related domains (multiple target domains). This setting differs from most previous studies (Blitzer et al., 2006) on DA which have attempted to design a specialized system for every specific target domain. In our view, although this setting is more challenging, it is more practical for RE. In fact, this setting can benefit considerably from our general approach of applying word representations and regularization. Finally, due to this setting, the best way to set up the regularization parameter is to impose the same regularization parameter on every feature rather than a skewed regularization (Jiang and Zhai, 2007b).

3 Related Work

Although word embeddings have been successfully employed in many NLP tasks (Collobert and Weston, 2008; Turian et al., 2010; Maas and Ng, 2010), the application of word embeddings in RE is very recent. Kuksa et al. (2010) propose an abstraction-augmented string kernel for bio-relation extraction via word embeddings. In the surge of deep learning, Socher et al. (2012) and Khashabi (2013) use pre-trained word embeddings as input for Matrix-Vector Recursive Neural Networks (MV-RNN) to learn compositional structures for RE. However, none of these works evaluate word embeddings for domain adaptation of RE which is our main focus in this paper.

Regarding domain adaptation, in representation learning, Blitzer et al. (2006) propose structural correspondence learning (SCL) while Huang and Yates (2010) attempt to learn a multi-dimensional feature representation. Unfortunately, these methods require unlabeled target domain data which are unavailable in our single-system setting of DA. Daumé III (2007) proposes an easy adaptation framework (EA) which is later extended to a semi-supervised version (EA++) to incorporate unlabeled

¹domain overfitting (Jiang and Zhai, 2007b)

beled data (Daumé III et al., 2010). In terms of word embeddings for DA, recently, Xiao and Guo (2013) present a log-bilinear language adaptation framework for sequential labeling tasks. However, these methods assume some labeled data in target domains and are thus not applicable in our setting of unsupervised DA. Above all, we move one step further by evaluating the effectiveness of word embeddings on domain adaptation for RE which is very different from the principal topic of sequence labeling in the previous research.

4 Word Representations

We consider two types of word representations and use them as additional features in our DA system, namely Brown word clustering (Brown et al., 1992) and word embeddings (Bengio et al., 2001). While word clusters can be recognized as an one-hot vector representation over a small vocabulary, word embeddings are dense, low-dimensional, and real-valued vectors (distributed representations). Each dimension of the word embeddings expresses a latent feature of the words, hopefully reflecting useful semantic and syntactic regularities (Turian et al., 2010). We investigate word embeddings induced by two typical language models: Collobert and Weston (2008) embeddings (C&W) (Collobert and Weston, 2008; Turian et al., 2010) and Hierarchical log-bilinear embeddings (HLBL) (Mnih and Hinton, 2007; Mnih and Hinton, 2009; Turian et al., 2010).

5 Feature Set

5.1 Baseline Feature Set

Sun et al. (2011) utilize the full feature set from (Zhou et al., 2005) plus some additional features and achieve the state-of-the-art feature-based RE system. Unfortunately, this feature set includes the *human-annotated* (gold-standard) information on entity and mention types which is often missing or noisy in reality (Plank and Moschitti, 2013). This issue becomes more serious in our setting of single-system DA where we have a single source domain with multiple dissimilar target domains and an automatic system able to recognize entity and mention types very well in different domains may not be available. Therefore, following the settings of Plank and Moschitti (2013), we will only assume entity boundaries and not rely on the gold standard information in the experiments. We apply the same feature set as Sun et al. (2011) but

remove the entity and mention type information².

5.2 Lexical Feature Augmentation

While Sun et al. (2011) show that adding word clusters to the heads of the two mentions is the most effective way to improve the generalization accuracy, the right lexical features into which word embeddings should be introduced to obtain the best adaptability improvement are unexplored. Also, which dimensionality of which word embedding should we use with which lexical features? In order to answer these questions, following Sun et al. (2011), we first group lexical features into 4 groups and rank their importance based on linguistic intuition and illustrations of the contributions of different lexical features from various feature-based RE systems. After that, we evaluate the effectiveness of these lexical feature groups for word embedding augmentation individually and incrementally according to the rank of importance. For each of these group combinations, we assess the system performance with different numbers of dimensions for both C&W and HLBL word embeddings. Let M1 and M2 be the first and second mentions in the relation. Table 1 describes the lexical feature groups.

Rank	Group	Lexical Features
1	HM	HM1 (head of M1)
		HM2 (head of M2)
2	BagWM	WM1 (words in M1)
		WM2 (words in M2)
3	HC	heads of chunks in context
4	BagWC	words of context

Table 1: Lexical feature groups ordered by importance.

6 Experiments

6.1 Tools and Data

Our relation extraction system is hierarchical (Bunescu and Mooney, 2005b; Sun et al., 2011) and apply maximum entropy (MaxEnt) in the MALLET³ toolkit as the machine learning tool. For Brown word clusters, we directly apply the clustering trained by Plank and Moschitti (2013)

²We have the same observation as Plank and Moschitti (2013) that when the gold-standard labels are used, the impact of word representations is limited since the gold-standard information seems to dominate. However, whenever the gold labels are not available or inaccurate, the word representations would be useful for improving adaptability performance. Moreover, in all the cases, regularization methods are still effective for domain adaptation of RE.

³<http://mallet.cs.umass.edu/>

System	In-domain (bn+nw)					Out-of-domain (bc development set)				
	C&W,25	C&W,50	C&W,100	HLBL,50	HLBL,100	C&W,25	C&W,50	C&W,100	HLBL,50	HLBL,100
1 Baseline	51.4	51.4	51.4	51.4	51.4	49.0	49.0	49.0	49.0	49.0
2 1+HM_ED	54.0(+2.6)	54.1(+2.7)	55.7(+4.3)	53.7(+2.3)	55.2(+3.8)	51.5(+2.5)	52.7(+3.7)	52.5(+3.5)	50.2(+1.2)	50.6(+1.6)
3 1+BagWM_ED	52.3(+0.9)	50.9(-0.5)	51.5(+0.1)	51.8(+0.4)	52.5(+1.1)	48.5(-0.5)	48.9(-0.1)	48.6(-0.4)	48.7(-0.3)	49.0(+0.0)
4 1+HC_ED	51.3(-0.1)	50.9(-0.5)	48.3(-3.1)	50.8(-0.6)	49.8(-1.6)	44.9(-4.1)	45.8(-3.2)	45.8(-3.2)	48.7(-0.3)	47.3(-1.7)
5 1+BagWC_ED	51.5(+0.1)	50.8(-0.6)	49.5(-1.9)	51.4(+0.0)	50.3(-1.1)	48.3(-0.7)	46.3(-2.7)	44.0(-5.0)	46.6(-2.4)	44.8(-4.2)
6 2+BagWM_ED	54.3(+2.9)	53.2(+1.8)	53.2(+1.8)	54.0(+2.6)	53.8(+2.4)	52.5(+3.5)	51.4(+2.4)	50.6(+1.6)	50.0(+1.0)	48.6(-0.4)
7 6+HC_ED	53.4(+2.0)	52.3(+0.9)	52.7(+1.3)	54.2(+2.8)	53.1(+1.7)	50.5(+1.5)	50.9(+1.9)	48.4(-0.6)	50.0(+1.0)	48.9(-0.1)
8 7+BagWC_ED	53.4(+2.0)	52.2(+0.8)	50.8(-0.6)	53.5(+2.1)	53.6(+2.2)	49.2(+0.2)	50.7(+1.7)	49.2(+0.2)	47.9(-1.1)	49.5(+0.5)

Table 2: In-domain and Out-of-domain performance for different embedding features. The cells in bold are the best results.

to facilitate system comparison later. We evaluate C&W word embeddings with 25, 50 and 100 dimensions as well as HLBL word embeddings with 50 and 100 dimensions that are introduced in Turian et al. (2010) and can be downloaded here⁴. The fact that we utilize the large, general and unbiased resources generated from the previous works for evaluation not only helps to verify the effectiveness of the resources across different tasks and settings but also supports our setting of single-system DA.

We use the ACE 2005 corpus for DA experiments (as in Plank and Moschitti (2013)). It involves 6 relation types and 6 domains: broadcast news (bn), newswire (nw), broadcast conversation (bc), telephone conversation (cts), weblogs (wl) and usenet (un). We follow the standard practices on ACE (Plank and Moschitti, 2013) and use **news (the union of bn and nw)** as the source domain and **bc, cts and wl** as our target domains. We take half of bc as the only target development set, and use the remaining data and domains for testing purposes (as they are small already). As noted in Plank and Moschitti (2013), the distributions of relations as well as the vocabularies of the domains are quite different.

6.2 Evaluation of Word Embedding Features

We investigate the effectiveness of word embeddings on lexical features by following the procedure described in Section 5.2. We test our system on two scenarios: In-domain: the system is trained and evaluated on the source domain (bn+nw, 5-fold cross validation); Out-of-domain: the system is trained on the source domain and evaluated on the target development set of bc (bc dev). Table 2 presents the F measures of this experiment⁵ (the

suffix *ED* in lexical group names is to indicate the embedding features).

From the tables, we find that for C&W and HLBL embeddings of 50 and 100 dimensions, the most effective way to introduce word embeddings is to add embeddings to the heads of the two mentions (row 2; both in-domain and out-of-domain) although it is less pronounced for HLBL embedding with 50 dimensions. Interestingly, for C&W embedding with 25 dimensions, adding the embedding to both heads and words of the two mentions (row 6) performs the best for both in-domain and out-of-domain scenarios. This is new compared to the word cluster features where the heads of the two mentions are always the best places for augmentation (Sun et al., 2011). It suggests that a suitable amount of embeddings for words in the mentions might be useful for the augmentation of the heads and inspires further exploration. Introducing embeddings to words of mentions alone has mild impact while it is generally a bad idea to augment chunk heads and words in the contexts.

Comparing C&W and HLBL embeddings is somehow more complicated. For both in-domain and out-of-domain settings with different numbers of dimensions, C&W embedding outperforms HLBL embedding when only the heads of the mentions are augmented while the degree of negative impact of HLBL embedding on chunk heads as well as context words seems less serious than C&W’s. Regarding the incremental addition of features (rows 6, 7, 8), C&W is better for the out-of-domain performance when 50 dimensions are used, whereas HLBL (with both 50 and 100 dimensions) is more effective for the in-domain setting. For the next experiments, we will apply the C&W embedding of 50 dimensions to the heads of the mentions for its best out-of-domain performance.

⁴<http://metaoptimize.com/projects/wordreprs/>

⁵All the in-domain improvement in rows 2, 6, 7 of Table 2 are significant at confidence levels $\geq 95\%$.

6.3 Domain Adaptation with Word Embeddings

This section examines the effectiveness of word representations for RE across domains. We evaluate word cluster and embedding (denoted by ED) features by adding them individually as well as simultaneously into the baseline feature set. For word clusters, we experiment with two possibilities: (i) only using a single prefix length of 10 (as Plank and Moschitti (2013) did) (denoted by WC10) and (ii) applying multiple prefix lengths of 4, 6, 8, 10 together with the full string (denoted by WC). Table 3 presents the system performance (F measures) for both in-domain and out-of-domain settings.

System	In-domain	bc	cts	wl
Baseline(B)	51.4	49.7	41.5	36.6
B+WC10	52.3(+0.9)	50.8(+1.1)	45.7(+4.2)	39.6(+3)
B+WC	53.7(+2.3)	52.8(+3.1)	46.8(+5.3)	41.7(+5.1)
B+ED	54.1(+2.7)	52.4(+2.7)	46.2(+4.7)	42.5(+5.9)
B+WC+ED	55.5(+4.1)	53.8(+4.1)	47.4(+5.9)	44.7(+8.1)

Table 3: Domain Adaptation Results with Word Representations. All the improvements over the baseline in Table 3 are significant at confidence level $\geq 95\%$.

The key observations from the table are:

(i): The baseline system achieves a performance of 51.4% within its own domain while the performance on target domains bc, cts, wl drops to 49.7%, 41.5% and 36.6% respectively. Our baseline performance is worse than that of Plank and Moschitti (2013) only on the target domain cts and better in the other cases. This might be explained by the difference between our baseline feature set and the feature set underlying their kernel-based system. However, the performance order across domains of the two baselines are the same. Besides, the baseline performance is improved over all target domains when the system is enriched with word cluster features of the 10 prefix length only (row 2).

(ii): Over all the target domains, the performance of the system augmented with word cluster features of various granularities (row 3) is superior to that when only cluster features for the prefix length 10 are added (row 2). This is significant (at confidence level $\geq 95\%$) for domains bc and wl and verifies our assumption that various granularities for word cluster features are more effective than a single granularity for domain adaptation of RE.

(iii): Row 4 shows that word embedding itself is also very useful for domain adaptation in RE since

it improves the baseline system for all the target domains.

(iv): In row 5, we see that the addition of both word cluster and word embedding features improves the system further and results in the best performance over all target domains (this is significant with confidence level $\geq 95\%$ in domains bc and wl). The result suggests that word embeddings seem to capture different information from word clusters and their combination would be effective to generalize relation extractors across domains. However, in domain cts, the improvement that word embeddings provide for word clusters is modest. This is because the RCV1 corpus used to induce the word embeddings (Turian et al., 2010) does not cover spoken language words in cts very well.

(v): Finally, the in-domain performance is also improved consistently demonstrating the robustness of word representations (Plank and Moschitti, 2013).

6.4 Domain Adaptation with Regularization

All the experiments we have conducted so far do not apply regularization for training. In this section, in order to evaluate the effect of regularization on the generalization capacity of relation extractors across domains, we replicate all the experiments in Section 6.3 but apply regularization when relation extractors are trained⁶. Table 4 presents the results.

System	In-domain	bc	cts	wl
Baseline(B)	56.2	55.5	48.7	42.2
B+WC10	57.5(+1.3)	57.3(+1.8)	52.3(+3.6)	45.0(+2.8)
B+WC	58.9(+2.7)	58.4(+2.9)	52.8(+4.1)	47.3(+5.1)
B+ED	58.9(+2.7)	59.5(+4.0)	52.6(+3.9)	48.6(+6.4)
B+WC+ED	59.4(+3.2)	59.8(+4.3)	52.9(+4.2)	49.7(+7.5)

Table 4: Domain Adaptation Results with Regularization. All the improvements over the baseline in Table 4 are significant at confidence level $\geq 95\%$.

For this experiment, every statement in (ii), (iii), (iv) and (v) of Section 6.3 also holds. More importantly, the performance in every cell of Table 4 is significantly better than the corresponding cell in Table 3 (5% or better gain in F measure, a significant improvement at confidence level $\geq 95\%$). This demonstrates the effectiveness of regularization for RE in general and for domain adaptation of RE specifically.

⁶We use a L2 regularizer with the regularization parameter of 0.5 for its best experimental results.

References

- Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2001. *A Neural Probabilistic Language Model*. In Advances in Neural Information Processing Systems (NIPS'13), pages 932-938, MIT Press, 2001.
- Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. *A Neural Probabilistic Language Model*. In Journal of Machine Learning Research (JMLR), 3, pages 1137-1155, 2003.
- 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, Sydney, Australia.
- John Blitzer, Mark Dredze, and Fernando Pereira. 2007. *Biographies, Bollywood, Boom-boxes, and Blenders: Domain Adaptation for Sentiment Classification*. In Proceedings of the ACL, pages 440-447, Prague, Czech Republic, June 2007.
- John Blitzer, Dean Foster, and Sham Kakade. 2011. *Domain Adaptation with Coupled Subspaces*. In Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, pages 173-181, Fort Lauderdale, FL, USA.
- Elizabeth Boschee, Ralph Weischedel, and Alex Zamarian. 2005. *Automatic Information Extraction*. In Proceedings of the International Conference on Intelligence Analysis.
- Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. *Class-Based n-gram Models of Natural Language*. In Journal of Computational Linguistics, Volume 18, Issue 4, pages 467-479, December 1992.
- Razvan C. Bunescu and Raymond J. Mooney. 2005a. *A Shortest Path Dependency Kernel for Relation Extraction*. In Proceedings of HLT/EMNLP.
- Razvan C. Bunescu and Raymond J. Mooney. 2005b. *Subsequence Kernels for Relation Extraction*. In Proceedings of NIPS.
- Yee S. Chan and Dan Roth. 2010. *Exploiting Background Knowledge for Relation Extraction*. In Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 152-160, Beijing, China, August.
- Ronan Collobert and Jason Weston. 2008. *A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning*. In International Conference on Machine Learning, ICML, 2008.
- Hal Daumé III. 2007. *Frustratingly Easy Domain Adaptation*. In Proceedings of the ACL, pages 256-263, Prague, Czech Republic, June 2007.
- Hal Daumé III, Abhishek Kumar and Avishek Saha. 2010. *Co-regularization Based Semi-supervised Domain Adaptation*. In Advances in Neural Information Processing Systems 23 (2010).
- Ralph Grishman, David Westbrook and Adam Meyers. 2005. *NYU's English ACE 2005 System Description*. ACE 2005 Evaluation Workshop.
- Fei Huang and Alexander Yates. 2010. *Exploring Representation-Learning Approaches to Domain Adaptation*. In Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing, pages 23-30, Uppsala, Sweden, July 2010.
- Jing Jiang and ChengXiang Zhai. 2007a. *A Systematic Exploration of the Feature Space for Relation Extraction*. In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'07), pages 113-120, 2007.
- Jing Jiang and ChengXiang Zhai. 2007b. *A Two-stage Approach to Domain Adaptation for Statistical Classifiers*. In Proceedings of the ACM 16th Conference on Information and Knowledge Management (CIKM'07), pages 401-410, 2007.
- Nanda Kambhatla. 2004. *Combining Lexical, Syntactic, and Semantic Features with Maximum Entropy Models for Information Extraction*. In Proceedings of ACL-04.
- Daniel Khashabi. 2013. *On the Recursive Neural Networks for Relation Extraction and Entity Recognition*. Technical Report (May, 2013), UIUC.
- Pavel Kuksa, Yanjun Qi, Bing Bai, Ronan Collobert, Jason Weston, Vladimir Pavlovic, and Xia Ning. 2010. *Semi-Supervised Abstraction-Augmented String Kernel for Multi-Level Bio-Relation Extraction*. In Proceedings of the 2010 European Conference on Machine Learning and Knowledge Discovery in Databases, Part II (ECML PKDD'10), pages 128-144, 2010.
- Andrew L. Maas and Andrew Y. Ng. 2010. *A Probabilistic Model for Semantic Word Vectors*. In NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2010.
- Andriy Mnih and Geoffrey Hinton. 2007. *Three new Graphical Models for Statistical Language Modelling*. In Proceedings of ICML'07, pages 641-648, Corvallis, OR, 2007.
- Andriy Mnih and Geoffrey Hinton. 2009. *A Scalable Hierarchical Distributed Language Model*. In NIPS, page 1081-1088.
- Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2009. *Convolution Kernels on Constituent, Dependency and Sequential Structures for Relation Extraction*. In Proceedings of EMNLP 09, pages 1378-1387, Stroudsburg, PA, USA.

- Barbara Plank and Alessandro Moschitti. 2013. *Embedding Semantic Similarity in Tree Kernels for Domain Adaptation of Relation Extraction*. In Proceedings of the ACL 2013, pages 1498-1507, Sofia, Bulgaria.
- Longhua Qian, Guodong Zhou, Qiaoming Zhu and Peide Qian. 2008. *Exploiting Constituent Dependencies for Tree Kernel-based Semantic Relation Extraction*. In Proceedings of COLING, pages 697-704, Manchester.
- Tobias Schnabel and Hinrich Schütze. 2014. *FLORS: Fast and Simple Domain Adaptation for Part-of-Speech Tagging*. In Transactions of the Association for Computational Linguistics, 2 (2014), pages 1526.
- Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. *Semantic Compositionality through Recursive Matrix-Vector Spaces*. In Proceedings EMNLP-CoNLL'12, pages 1201-1211, Jeju Island, Korea, July 2012.
- Ang Sun, Ralph Grishman, and Satoshi Sekine. 2011. *Semi-supervised Relation Extraction with Large-scale Word Clustering*. In Proceedings of ACL-HLT, pages 521-529, Portland, Oregon, USA.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. *Word representations: A simple and general method for semi-supervised learning*. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10), pages 384-394, Uppsala, Sweden, July, 2010.
- Min Xiao and Yuhong Guo. 2013. *Domain Adaptation for Sequence Labeling Tasks with a Probabilistic Language Adaptation Model*. In Proceedings of the 30th International Conference on Machine Learning (ICML-13), pages 293-301, 2013.
- Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. *Kernel Methods for Relation Extraction*. Journal of Machine Learning Research, 3:1083-1106.
- Min Zhang, Jie Zhang, Jian Su, and GuoDong Zhou. 2006. *A Composite Kernel to Extract Relations between Entities with both Flat and Structured Features*. In Proceedings of COLING-ACL-06, pages 825-832, Sydney.
- Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. *Exploring various Knowledge in Relation Extraction*. In Proceedings of ACL'05, pages 427-434, Ann Arbor, USA, 2005.

Graph Ranking for Collective Named Entity Disambiguation

Ayman Alhelbawy^{1,2} and Robert Gaizauskas¹

¹The University of Sheffield, Regent Court, 211 Portobello Street, Sheffield, S1 4DP, U.K

²Faculty of Computers and Information, Fayoum University, Fayoum, Egypt

ayman, R.Gaizauskas@dcs.shef.ac.uk

Abstract

Named Entity Disambiguation (NED) refers to the task of mapping different named entity mentions in running text to their correct interpretations in a specific knowledge base (KB). This paper presents a collective disambiguation approach using a graph model. All possible NE candidates are represented as nodes in the graph and associations between different candidates are represented by edges between the nodes. Each node has an initial confidence score, e.g. entity popularity. Page-Rank is used to rank nodes and the final rank is combined with the initial confidence for candidate selection. Experiments on 27,819 NE textual mentions show the effectiveness of using Page-Rank in conjunction with initial confidence: 87% accuracy is achieved, outperforming both baseline and state-of-the-art approaches.

1 Introduction

Named entities (NEs) have received much attention over the last two decades (Nadeau and Sekine, 2007), mostly focused on recognizing the boundaries of textual NE mentions and classifying them as, e.g., Person, Organization or Location. However, references to entities in the real world are often ambiguous: there is a many-to-many relation between NE mentions and the entities they denote in the real world. For example, *Norfolk* may refer to a person, “Peter Norfolk, a wheelchair tennis player”, a place in the UK, “Norfolk County”, or in the US, “Norfolk, Massachusetts”; conversely, one entity may be known by many names, such as “Cat Stevens”, “Yusuf Islam” and “Steven Georgiou”. The NED task is to establish a correct mapping between each NE mention in a document and the real world entity it denotes. Following most researchers in this area, we treat entries in a large

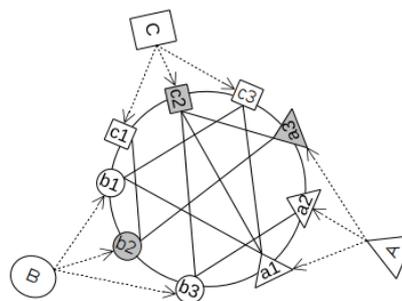


Figure 1: Example of solution graph

knowledge base (KB) as surrogates for real world entities when carrying out NED and, in particular, use Wikipedia as the reference KB for disambiguating NE mentions. NED is important for tasks like KB population, where we want to extract new information from text about an entity and add this to a pre-existing entry in a KB; or for information retrieval, where we may want to cluster or filter results for different entities with the same textual mentions.

The main hypothesis in this work is that different NEs in a document help to disambiguate each other. The problem is that other textual mentions in the document are also ambiguous. So, what is needed is a *collective disambiguation* approach that jointly disambiguates all NE textual mentions.

In our approach we model each possible candidate for every NE mention in a document as a distinct node in a graph and model candidate coherence by links between the nodes. We call such graphs *solution graphs*. Figure 1 shows an example of the solution graph for three mentions “A”, “B”, and “C” found in a document, where the candidate entities for each mention are referred to using the lower case form of the mention’s letter together with a distinguishing subscript. The goal of disambiguation is to find a set of nodes where only one candidate is selected from the set of entities associated with each mention, e.g. a_3 , b_2 , c_2 .

Our approach first ranks all nodes in the solution graph using the Page-Rank algorithm, then re-

ranks all nodes by combining the initial confidence and graph ranking scores. We consider several different measures for computing the initial confidence assigned to each node and several measures for determining and weighting the graph edges. Node linking relies on the fact that the textual portion of KB entries typically contains mentions of other NEs. When these mentions are hyper-linked to KB entries, we can infer that there is some relation between the real world entities corresponding to the KB entries, i.e. that they should be linked in our solution graph. These links also allow us to build up statistical co-occurrence counts between entities that occur in the same context which may be used to weight links in our graph.

We evaluate our approach on the AIDA dataset (Hoffart et al., 2011). Comparison with the baseline approach and some state-of-the-art approaches shows our approach offers substantial improvements in disambiguation accuracy.

2 Related Work

In 2009, NIST proposed the shared task challenge of Entity Linking (EL) (McNamee and Dang, 2009). EL is a similar but broader task than NED because NED is concerned with disambiguating a textual NE mention where the correct entity is known to be one of the KB entries, while EL also requires systems to deal with the case where there is no entry for the NE in the reference KB. Ji et al. (2011) group and summarise the different approaches to EL taken by participating systems.

In general, there are two main lines of approach to the NED problem. *Single entity disambiguation approaches (SNED)*, disambiguate one entity at a time without considering the effect of other NEs. These approaches use local context textual features of the mention and compare them to the textual features of NE candidate documents in the KB, and link to the most similar. The first approach in this line was Bunescu and Pasca (2006), who measure similarity between the textual context of the NE mention and the Wikipedia categories of the candidate. More similarity features were added by Cucerzan (2007) who realized that topical coherence between a candidate entity and other entities in the context will improve NED accuracy and by Milne and Witten (2008) who built on Cucerzan’s work. Han and Sun (2011) combine different forms of disambiguation knowledge using evidence from mention-entity associations and

entity popularity in the KB, and context similarity.

The second line of approach is *collective named entity disambiguation (CNED)*, where all mentions of entities in the document are disambiguated jointly. These approaches try to model the interdependence between the different candidate entities for different NE mentions in the query document, and reformulate the problem of NED as a global optimization problem whose aim is to find the best set of entities. As this new formulation is NP-hard, many approximations have been proposed. Alhelbawy and Gaizauskas (2013) proposed a sequence dependency model using HMMs to model NE interdependency. Another approximation uses a mixture of local and global features to train the coefficients of a linear ranking SVM to rank different NE candidates (Ratinov et al., 2011). Shirakawa et al. (2011) cluster related textual mentions and assign a concept to each cluster using a probabilistic taxonomy. The concept associated with a mention is used in selecting the correct entity from the Freebase KB.

Graph models are widely used in collective approaches¹. All these approaches model NE interdependencies, while different methods may be used for disambiguation. Han (2011) uses local dependency between NE mention and the candidate entity, and semantic relatedness between candidate entities to construct a referent graph, proposing a collective inference algorithm to infer the correct reference node in the graph. Hoffert (2011) poses the problem as one of finding a dense sub-graph, which is infeasible in a huge graph. So, an algorithm originally used to find strongly interconnected, size-limited groups in social media is adopted to prune the graph, and then a greedy algorithm is used to find the densest graph.

Our proposed model uses the Page-Rank (PR) algorithm (Page et al., 1999), which to our knowledge has not previously been applied to NED. Xing and Ghorbani (2004) adopted PR to consider the weights of links and the nodes’ importance. PR and Personalized PR algorithms have been used successfully in WSD (Sinha and Mihalcea, 2007; Agirre and Soroa, 2009).

3 Solution Graph

In this section we discuss the construction of a graph representation that we call the *solution*

¹Graph models are also widely used in Word Sense Disambiguation (WSD), which has lots of similarities to NED (Gutiérrez et al., 2011; Gutiérrez et al., 2012).

graph. The input is a document containing pre-tagged NE textual mentions. The solution graph is an undirected graph $G = (V, D)$ where V is the node set of all possible NE candidates for different textual mentions in the input document and D is the set of edges between nodes. Edges are not drawn between different nodes for the same mention. They are drawn between two entities when there is a relation between them, as described below. Each candidate has associated with it an initial confidence score, also detailed below.

Assume the input document D has a set of mentions $M = \{m_1, m_2, m_3, \dots, m_k\}$. For each $m_i \in M$, we rank each candidate entity, where the list of candidates for a mention m_i is $E_i = \{e_{i,1}, e_{i,2}, \dots, e_{i,j}\}$. The graph nodes are formulated as a set $V = \{(m_i, e_{i,j}) \mid \forall e_{i,j} \in E_i, \forall m_i \in M\}$. Nodes are represented as ordered pairs of textual mentions and candidate entities, since the same entity may be found multiple times as a candidate for different textual mentions and each occurrence must be evaluated independently.

3.1 NE Candidate Generation

The first step in constructing a solution graph is to find all possible candidates for each NE mention in the query document. For each such mention the KB entry titles are searched to find all entries to which the mention could refer. This includes entries with titles that fully or partially contain the query mention and those that could be an acronym of the query mention. These candidate entries are paired with their textual mentions in the document to become nodes in the solution graph.

3.2 Initial Confidence

Initial confidence $IConf(e_{i,j})$ is an independent feature of the NE candidate regardless of other candidates in the document. This confidence may be calculated locally using the local mention context, or globally using, e.g., the Freebase popularity score for the KB entry (Bollacker et al., 2008).

Local NE Candidate Confidence: The local confidence is computed by a similarity measure between the NE mention in the query document and the KB entry of the candidate entity. We propose four different measures to be used in the disambiguation phase.

cos: The cosine similarity between the named entity textual mention and the KB entry title.

jwSim: While the cosine similarity between a textual mention in the document and the candidate

NE title in the KB is widely used in NED, this similarity is a misleading feature. For example, the textual mention “Essex” may refer to either of the following candidates “Essex County Cricket Club” or “Danbury, Essex”, both of which are returned by the candidate generation process. The cosine similarity between “Essex” and “Danbury, Essex” is higher than that between “Essex” and “Essex County Cricket Club”, which is not helpful in the NED setting. We adopted a new mention-candidate similarity function, $jwSim$, which uses Jaro-Winkler similarity as a first estimate of the initial confidence value for each candidate. This function considers all terms found in the candidate entity KB entry title, but not in the textual mention as disambiguation terms. The percentage of disambiguation terms found in the query document is used to boost in the initial $jwSim$ value, in addition to an acronym check (whether the NE textual mention could be an acronym for a specific candidate entity title). Experiments show that $jwSim$ performs much better than cos .

ctxt: The cosine similarity between the sentence containing the NE mention in the query document and the textual description of the candidate NE in the KB (we use the first section of the Wikipedia article as the candidate entity description).

Global NE Candidate Confidence: Global confidence is a measure of the global importance of the candidate entity. Entity popularity has been used successfully as a discriminative feature for NED (Nebhi, 2013). Freebase provides an API to get an entity’s popularity score (**FB**), which is computed during Freebase indexing. This score is a function of the entity’s inbound and outbound link counts in Freebase and Wikipedia². The initial confidence is not normalized across all NEs because each score is calculated independently. Initial confidence scores of all candidates for a single NE mention are normalized to sum to 1.

3.3 Entity Coherence

Entity coherence refers to the real world relatedness of different entities which are candidate interpretations of different textual mentions in the document. It is not based on context, so it is always the same regardless of the query document. Coherence is represented as an edge between nodes in the solution graph. We used two measures for coherence, described as follows:

²<https://developers.google.com/freebase/v1/search>

Ref: Uses the Wikipedia documents for both entity candidates to check if either document has a link to the other. This relation is directed, but we assume an inverse relation also exists; so this relation is represented as undirected.

$$\text{Ref}(e_i, e_j) = \begin{cases} 1, & \text{if } e_i \text{ or } e_j \text{ refers to the other} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

JProb: An estimate of the probability of both entities appearing in the same sentence. Wikipedia documents are used to estimate this probability, as shown in (2), where $S(e)$ is the set of all sentences that contain the entity e and S the set of sentences containing any entity references.

$$\text{JProb}(e_i, e_j) = \frac{|S(e_i) \cap S(e_j)|}{|S|} \quad (2)$$

4 Disambiguation

The solution graph contains all possible candidates for each NE mention in the document. Each candidate has an initial confidence, with some connected by association relations. The disambiguation phase ranks all nodes in the solution graph and selects the best from the candidate list for each NE textual mention. The process of disambiguation consists of three steps. The first step is initial graph ranking, where all nodes are ranked according to the link structure. The second step is to re-rank the nodes by combining the graph rank with the initial confidence. The highest rank is not always correct, so in the third step a selection algorithm is used to choose the best candidate.

Graph Ranking: The links between different candidates in the solution graph represent real world relations. These relations may be used to reliably boost relevant candidates. All nodes in the graph are ranked according to these relations using PR. Initial confidence is used as an initial rank for the graph nodes, while entities' coherence measures are used as link weights which play a role in distributing a node's rank over its outgoing nodes.

Candidate Re-ranking: A problem with Page-Rank for our purposes is the dissipation of initial node weight (confidence) over all outgoing nodes. The final rank of a node is based solely on the importance of incoming nodes and the initial confidence play no further role. In our case this is not appropriate, so the final rank for each mention is determined after graph ranking, by combining the graph rank with the initial confidence.

Let us refer to the graph rank of a candidate as $PR(e_i)$. Two combination schemes are used:

$$R_s(e_{i,j}) = IConf(e_{i,j}) + PR(e_{i,j}) \quad (3)$$

$$R_m(e_{i,j}) = IConf(e_{i,j}) \times PR(e_{i,j}) \quad (4)$$

Named Entity Selection: The simplest approach is to select the highest ranked entity in the list for each mention m_i according to equation 5, where R could refer to R_m or R_s . However, we found that a dynamic choice between the re-ranking schemes, based on the difference between the top two candidates, as described in algorithm 1 and indicated by e^g , works best. The underlying intuition of this algorithm is that a greater difference between the top ranks reflects more confident discrimination between candidates. So, the two combination schemes assign different ranks to the candidates and the algorithm selects the scheme which appears more discriminative.

$$\hat{e}_i = \underset{e_{i,j}}{\text{argmax}} R(e_{i,j}) \quad (5)$$

Data: Two lists, R1 and R2, of candidates E_i , where R1 is ranked using R_s , and R2 is ranked using R_m

Result: One NE e_i^g

Sort R1 and R2 in descending order;

R1diff = R1[0]-R1[1];

R2diff = R2[0]-R2[1];

if R1diff > R2diff **then**

 | return highest rank scored entity of R1

else

 | return highest rank scored entity of R2

end

Algorithm 1: Selection Algorithm

5 Experiments and Results

We used AIDA dataset³, which is based on the CoNLL 2003 data for NER tagging. All mentions are manually disambiguated against Wikipedia (Hoffart et al., 2011). This dataset contains 1393 documents and 34,965 annotated mentions. We only consider NE mentions with an entry in the Wikipedia KB, ignoring the 20% of query mentions (7136) without a link to the KB, as Hoffart did. Micro-averaged and macro-averaged accuracy are used for evaluation. In this context micro-averaged accuracy corresponds to the proportion of textual mentions correctly disambiguated while macro-averaged accuracy corresponds to the proportion of textual mentions correctly disambiguated per entity, averaged over all entities.

5.1 Results

Initially, we evaluated the performance of two baselines. One is a setup where a ranking based solely on different initial confidence scores is used

³<http://www.mpi-inf.mpg.de/yago-naga/aida/>

	<i>IConf</i>	<i>PR_C</i>	<i>PR_I</i>	<i>PR_{IC}</i>	Cucerzan	Kulkarni	Hoffart	Shirakawa	Alhelbawy
<i>A_{macro}</i>	78.09	80.98	84.19	82.80	43.74	76.74	81.91	83.02	74.18
<i>A_{micro}</i>	80.55	83.59	87.59	86.10	51.03	72.87	81.82	82.29	78.49

Table 1: Results comparison between Proposed Approach and State-of-the-art

<i>IConf</i>	<i>PR</i>		e^g	
	<i>A_{micro}</i>	<i>A_{macro}</i>	<i>A_{micro}</i>	<i>A_{macro}</i>
cos	70.6	60.83	78.41	72.35
jwSim	70.61	60.94	83.16	78.28
ctxt	70.61	60.83	75.45	65.22
freebase	71.78	81.07	87.59	84.19

Table 2: Results using initial confidence (PR_I)

Edge Weight	<i>PR</i>		e^g	
	<i>A_{micro}</i>	<i>A_{macro}</i>	<i>A_{micro}</i>	<i>A_{macro}</i>
<i>Jprob</i>	66.52	55.83	83.31	80.38
<i>Ref</i>	67.48	59.76	81.80	78.53
<i>prob + refs</i>	72.69	65.71	83.46	80.69

Table 3: Results using weighted edges (PR_C)

for candidate selection, i.e. without using PR. In this setup a ranking based on Freebase popularity does best, with micro- and macro-averaged accuracy scores of 80.55% and 78.09% respectively. This is a high baseline, close to the state-of-the-art. Our second baseline is the basic PR algorithm, where weights of nodes and edges are uniform (i.e. initial node and edge weights set to 1, edges being created wherever REF or JProb are not zero). Micro and macro accuracy scores of 70.60% and 60.91% were obtained with this baseline.

To study graph ranking using PR, and the contributions of the initial confidence and entity coherence, experiments were carried out using PR in different modes and with different selection techniques. In the first experiment, referred to as PR_I , initial confidence is used as an initial node rank for PR and edge weights are uniform, edges, as in the PR baseline, being created wherever REF or JProb are not zero. Table 2 shows the results both before re-ranking, i.e. using only the PR score for ranking, and after re-ranking using the dynamic selection scheme e^g . When comparing these results to the PR baseline we notice a slight positive effect when using the initial confidence as an initial rank instead of uniform ranking. The major improvement comes from re-ranking nodes by combining initial confidence with PR score.

In our second experiment, PR_C , entity coherence features are tested by setting the edge weights to the coherence score and using uniform initial node weights. We compared JProb and Ref

Edge Weight	e^g (jwSim)		e^g (freebase)	
	<i>A_{micro}</i>	<i>A_{macro}</i>	<i>A_{micro}</i>	<i>A_{macro}</i>
<i>Jprob</i>	82.56	76.16	86.29	82.77
<i>Ref</i>	78.61	71.12	83.16	80.01
<i>Jprob + Ref</i>	81.97	75.63	86.10	82.80

Table 4: Results using *IConf* and weighted edges PR_{IC}

edge weighting approaches, where for each approach edges were created only where the coherence score according to the approach was non-zero. We also investigated a variant, called JProb + Ref, in which the Ref edge weights are normalized to sum to 1 over the whole graph and then added to the JProb edge weights (here edges result whenever JProb or Ref scores are non-zero). Results in Table 3 show the *JProb* feature seems to be more discriminative than the *Ref* feature but the combined *Jprob + Ref* feature performs better than each separately, just outperforming the baseline. We used the best initial confidence score (Freebase) for re-ranking. Again, combining the initial confidence with the PR score improves the results.

Finally, Table 4 shows the accuracy when using different combinations of initial confidence and entity coherence scores just in the case when re-ranking is applied. Here the *jprob + refs* combination does not add any value over *jprob* alone. Interestingly using initial confidence with differentially weighted edges does not show any benefit over using initial confidence and uniformly weighted edges (Table 2).

To compare our results with the state-of-the-art, we report Hoffart et al.’s (2011) results as they re-implemented two other systems and also ran them over the AIDA dataset. We also compare with Alhelbawy and Gaizauskas (2013) and Shirakawa et al. (2011) who carried out their experiments using the same dataset. Table 1 presents a comparison between our approach and the state-of-the-art and shows our approach exceeds the state-of-the-art. Furthermore our approach is very simple and direct to apply, unlike Hoffart et al.’s and Shirakawa et al.’s which are considerably more complex. Also, our approach does not need any kind of training, as does the Alhelbawy approach.

6 Conclusion

Our results show that Page-Rank in conjunction with re-ranking by initial confidence score can be used as an effective approach to collectively disambiguate named entity textual mentions in a document. Our proposed features are very simple and easy to extract, and work well when employed in PR. In future work we plan to explore enriching the edges between nodes by incorporating semantic relations extracted from an ontology.

References

- Eneko Agirre and Aitor Soroa. 2009. Personalizing pagerank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–41. Association for Computational Linguistics.
- Ayman Alhelbawy and Robert Gaizauskas. 2013. Named entity disambiguation using hmms. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, volume 3, pages 159–162.
- 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, SIGMOD '08*, pages 1247–1250, New York, NY, USA. ACM.
- Razvan C. Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *EACL. The Association for Computer Linguistics*.
- Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of EMNLP-CoNLL*, volume 6, pages 708–716.
- Yoan Gutiérrez, Sonia Vázquez, and Andrés Montoyo. 2011. Word sense disambiguation: a graph-based approach using n-cliques partitioning technique. In *Natural Language Processing and Information Systems*, pages 112–124. Springer.
- Yoan Gutiérrez, Sonia Vázquez, and Andrés Montoyo. 2012. A graph-based approach to wsd using relevant semantic trees and n-cliques model. In *Computational Linguistics and Intelligent Text Processing*, pages 225–237. Springer.
- Xianpei Han and Le Sun. 2011. A generative entity-mention model for linking entities with knowledge base. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 945–954. Association for Computational Linguistics.
- Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.
- 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.
- Heng Ji and Ralph Grishman. 2011. Knowledge base population: successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1148–1158. Association for Computational Linguistics.
- Paul McNamee and Hoa Trang Dang. 2009. Overview of the tac 2009 knowledge base population track. In *Text Analysis Conference (TAC)*, volume 17, pages 111–113.
- David Milne and Ian H Witten. 2008. Learning to link with wikipedia. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 509–518. ACM.
- David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1):3–26.
- Kamel Nebhi. 2013. Named entity disambiguation using freebase and syntactic parsing. In CEUR-WS.org, editor, *Proceedings of the First International Workshop on Linked Data for Information Extraction (LD4IE 2013) co-located with the 12th International Semantic Web Conference (ISWC 2013)*.
- Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November. Previous number = SIDL-WP-1999-0120.
- Lev Ratinov, 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.
- Masumi Shirakawa, Haixun Wang, Yangqiu Song, Zhongyuan Wang, Kotaro Nakayama, Takahiro Hara, and Shojiro Nishio. 2011. Entity disambiguation based on a. Technical report, Technical report, Technical Report MSR-TR-2011-125, Microsoft Research.
- Ravi Sinha and Rada Mihalcea. 2007. Unsupervised graph-based word sense disambiguation using measures of word semantic similarity. In *Semantic Computing, 2007. ICSC 2007. International Conference on*, pages 363–369. IEEE.
- Wenpu Xing and Ali Ghorbani. 2004. Weighted pagerank algorithm. In *Communication Networks and Services Research, 2004. Proceedings. Second Annual Conference on*, pages 305–314. IEEE.

Descending-Path Convolution Kernel for Syntactic Structures

Chen Lin¹, Timothy Miller¹, Alvin Kho¹, Steven Bethard²,
Dmitriy Dligach¹, Sameer Pradhan¹ and Guergana Savova¹,

¹ Children’s Hospital Boston Informatics Program and Harvard Medical School
{firstname.lastname}@childrens.harvard.edu

² Department of Computer and Information Sciences, University of Alabama at Birmingham
bethard@cis.uab.edu

Abstract

Convolution tree kernels are an efficient and effective method for comparing syntactic structures in NLP methods. However, current kernel methods such as subset tree kernel and partial tree kernel understate the similarity of very similar tree structures. Although soft-matching approaches can improve the similarity scores, they are corpus-dependent and match relaxations may be task-specific. We propose an alternative approach called descending path kernel which gives intuitive similarity scores on comparable structures. This method is evaluated on two temporal relation extraction tasks and demonstrates its advantage over rich syntactic representations.

1 Introduction

Syntactic structure can provide useful features for many natural language processing (NLP) tasks such as semantic role labeling, coreference resolution, temporal relation discovery, and others. However, the choice of features to be extracted from a tree for a given task is not always clear. Convolution kernels over syntactic trees (tree kernels) offer a potential solution to this problem by providing relatively efficient algorithms for computing similarities between entire discrete structures. These kernels use tree fragments as features and count the number of common fragments as a measure of similarity between any two trees.

However, conventional tree kernels are sensitive to pattern variations. For example, two trees in Figure 1(a) sharing the same structure except for one terminal symbol are deemed at most 67% similar by the conventional tree kernel (PTK) (Moschitti, 2006). Yet one might expect a higher similarity given their structural correspondence.

The similarity is further attenuated by trivial structure changes such as the insertion of an ad-

jective in one of the trees in Figure 1(a), which would reduce the similarity close to zero. Such an abrupt attenuation would potentially propel a model to memorize training instances rather than generalize from trends, leading towards overfitting.

In this paper, we describe a new kernel over syntactic trees that operates on descending paths through the tree rather than production rules as used in most existing methods. This representation is reminiscent of Sampson’s (2000) leaf-ancestor paths for scoring parse similarities, but here it is generalized over all ancestor paths, not just those from the root to a leaf. This approach assigns more robust similarity scores (e.g., 78% similarity in the above example) than other soft matching tree kernels, is faster than the partial tree kernel (Moschitti, 2006), and is less *ad hoc* than the grammar-based convolution kernel (Zhang et al., 2007).

2 Background

2.1 Syntax-based Tree Kernels

Syntax-based tree kernels quantify the similarity between two constituent parses by counting their common sub-structures. They differ in their definition of the sub-structures.

Collins and Duffy (2001) use a subset tree (SST) representation for their sub-structures. In the SST representation, a subtree is defined as a subgraph with more than one node, in which only full production rules are expanded. While this approach is widely used and has been successful in many tasks, the production rule-matching constraint may be unnecessarily restrictive, giving zero credit to rules that have only minor structural differences. For example, the similarity score between the NPs in Figure 1(b) would be zero since the production rule is different (the overall similarity score is above-zero because of matching pre-terminals).

The partial tree kernel (PTK) relaxes the definition of subtrees to allow partial production rule

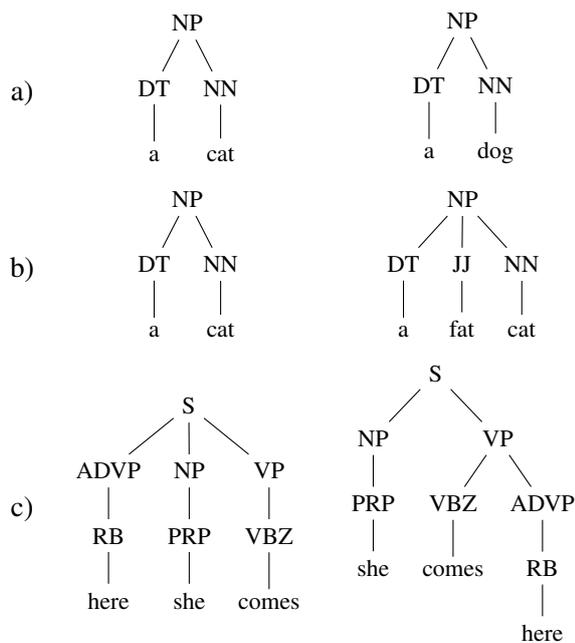


Figure 1: Three example tree pairs.

matching (Moschitti, 2006). In the PTK, a subtree may or may not expand any child in a production rule, while maintaining the ordering of the child nodes. Thus it generates a very large but sparse feature space. To Figure 1(b), the PTK generates fragments (i) [NP [DT a] [JJ fat]]; (ii) [NP [DT a] [NN cat]]; and (iii) [NP [JJ fat] [NN cat]], among others, for the second tree. This allows for partial matching – substructure (ii) – while also generating some fragments that violate grammatical intuitions.

Zhang et al. (2007) address the restrictiveness of SST by allowing soft matching of production rules. They allow partial matching of optional nodes based on the Treebank. For example, the rule $NP \rightarrow DT JJ NN$ indicates a noun phrase consisting of a determiner, adjective, and common noun. Zhang et al.’s method designates the JJ as optional, since the Treebank contains instances of a reduced version of the rule without the JJ node ($NP \rightarrow DT NN$). They also allow node matching among similar preterminals such as JJ, JJR, and JJS, mapping them to one equivalence class.

Other relevant approaches are the spectrum tree (SpT) (Kuboyama et al., 2007) and the route kernel (RtT) (Aiolli et al., 2009). SpT uses a q-gram – a sequence of connected vertices of length q – as their sub-structure. It observes grammar rules by recording the orientation of edges: $a \leftarrow b \rightarrow c$ is different from $a \rightarrow b \rightarrow c$. RtT uses a set of routes as basic structures, which observes grammar rules by

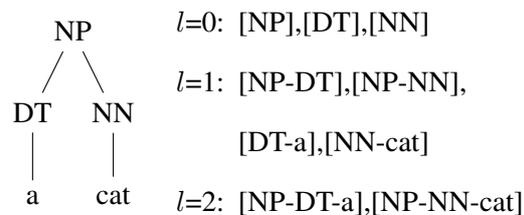


Figure 2: A parse tree (left) and its descending paths according to Definition 1 (l - length).

recording the index of a neighbor node.

2.2 Temporal Relation Extraction

Among NLP tasks that use syntactic information, temporal relation extraction has been drawing growing attention because of its wide applications in multiple domains. As subtasks in TempEval 2007, 2010 and 2013, multiple systems were built to create labeled links from events to events/timestamps by using a variety of features (Bethard and Martin, 2007; Llorens et al., 2010; Chambers, 2013). Many methods exist for synthesizing syntactic information for temporal relation extraction, and most use traditional tree kernels with various feature representations. Mirroshandel et al. (2009) used the path-enclosed tree (PET) representation to represent syntactic information for temporal relation extraction on the TimeBank (Pustejovsky et al., 2003) and the AQUAINT TimeML corpus¹. The PET is the smallest subtree that contains both proposed arguments of a relation. Hovy et al. (2012) used bag tree structures to represent the bag of words (BOW) and bag of part of speech tags (BOP) between the event and time in addition to a set of baseline features, and improved the temporal linking performance on the TempEval 2007 and Machine Reading corpora (Strassel et al., 2010). Miller et al. (2013) used PET tree, bag tree, and path tree (PT, which is similar to a PET tree with the internal nodes removed) to represent syntactic information and improved the temporal relation discovery performance on THYME data² (Styler et al., 2014). In this paper, we also use syntactic structure-enriched temporal relation discovery as a vehicle to test our proposed kernel.

3 Methods

Here we describe the Descending Path Kernel (DPK).

¹<http://www.timeml.org>

²<http://thyme.healthnlp.org>

Definition 1 (Descending Path): Let T be a parse tree, v any non-terminal node in T , dv a descendant of v , including terminals. A descending path is the sequence of indexes of edges connecting v and dv , denoted by $[v - \dots - dv]$. The length l of a descending path is the number of connecting edges. When $l = 0$, a descending path is the non-terminal node itself, $[v]$. Figure 2 illustrates a parse tree and its descending paths of different lengths.

Suppose that all descending paths of a tree T are indexed $1, \dots, n$, and $path_i(T)$ is the frequency of the i -th descending path in T . We represent T as a vector of frequencies of all its descending paths: $\Phi(T) = (path_1(T), \dots, path_n(T))$.

The similarity between any two trees T_1 and T_2 can be assessed via the dot product of their respective descending path frequency vector representations: $K(T_1, T_2) = \langle \Phi(T_1), \Phi(T_2) \rangle$.

Compared with the previous tree kernels, our descending path kernel has the following advantages: 1) the sub-structures are simplified so that they are more likely to be shared among trees, and therefore the sparse feature issues of previous kernels could be alleviated by this representation; 2) soft matching between two similar structures (e.g., NP→DT JJ NN versus NP→DT NN) have high similarity without reference to any corpus or grammar rules;

Following Collins and Duffy (2001), we derive a recursive algorithm to compute the dot product of the descending path frequency vector representations of two trees T_1 and T_2 :

$$\begin{aligned}
K(T_1, T_2) &= \langle \Phi(T_1), \Phi(T_2) \rangle \\
&= \sum_i path_i(T_1) \cdot path_i(T_2) \\
&= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \sum_i I_{path_i}(n_1) \cdot I_{path_i}(n_2) \\
&= \sum_{\substack{n_1 \in N_1 \\ n_2 \in N_2}} C(n_1, n_2)
\end{aligned} \tag{1}$$

where N_1 and N_2 are the sets of nodes in T_1 and T_2 respectively, i indexes the set of possible paths, $I_{path_i}(n)$ is an indicator function that is 1 iff the descending $path_i$ is rooted at node n or 0 otherwise. $C(n_1, n_2)$ counts the number of common descending paths rooted at nodes n_1 and n_2 :

$$C(n_1, n_2) = \sum_i I_{path_i}(n_1) \cdot I_{path_i}(n_2)$$

$C(n_1, n_2)$ can be computed in polynomial time by

the following recursive rules:

Rule 1: If n_1 and n_2 have different labels (e.g., "DT" versus "NN"), then $C(n_1, n_2) = 0$;

Rule 2: Else if n_1 and n_2 have the same labels and are both pre-terminals (POS tags), then

$$C(n_1, n_2) = 1 + \begin{cases} 1 & \text{if } term(n_1) = term(n_2) \\ 0 & \text{otherwise.} \end{cases}$$

where $term(n)$ is the terminal symbol under n ;

Rule 3: Else if n_1 and n_2 have the same labels and they are not both pre-terminals, then:

$$C(n_1, n_2) = 1 + \sum_{\substack{n_i \in children(n_1) \\ n_j \in children(n_2)}} C(n_i, n_j)$$

where $children(m)$ are the child nodes of m .

As in other tree kernel approaches (Collins and Duffy, 2001; Moschitti, 2006), we use a discount parameter λ to control for the disproportionately large similarity values of large tree structures. Therefore, Rule 2 becomes:

$$C(n_1, n_2) = 1 + \begin{cases} \lambda & \text{if } term(n_1) = term(n_2) \\ 0 & \text{otherwise.} \end{cases}$$

and Rule 3 becomes:

$$C(n_1, n_2) = 1 + \lambda \sum_{\substack{n_i \in children(n_1) \\ n_j \in children(n_2)}} C(n_i, n_j)$$

Note that Eq. (1) is a convolution kernel under the kernel closure properties described in Hausler (1999). Rules 1-3 show the equivalence between the number of common descending paths rooted at nodes n_1 and n_2 , and the number of matching nodes below n_1 and n_2 .

In practice, there are many non-matching nodes, and most matching nodes will have only a few matching children, so the running time, as in SST, will be approximated by the number of matching nodes between trees.

3.1 Relationship with other kernels

For a given tree, DPK will generate significantly fewer sub-structures than PTK, since it does not consider all ordered permutations of a production rule. Moreover, the fragments generated by DPK are more likely to be shared among different trees. For the number of corpus-wide fragments, it is

Kernel	ID	#Frag	Sim	N(Sim)
SST $O(\rho N_1 N_2)$	a	9	3	0.50
	b	15	2	0.25
	c	63	7	0.20
DPK $O(\rho^2 N_1 N_2)$	a	11	7	0.78
	b	13	9	0.83
	c	31	22	0.83
PTK $O(\rho^3 N_1 N_2)$	a	20	10	0.67
	b	36	15	0.65
	c	127	34	0.42

Table 1: Comparison of the worst case computational complexity (ρ - the maximum branching factor) and kernel performance on the 3 examples from Figure 1. $\#Frag$ is the number of fragments, $N(Sim)$ is the normalized similarity. Please see the online supplementary note for detailed fragments of example (a).

possible that $DPK \leq SST \leq PTK$. In Table 1, given $\lambda = 1$, we compare the performance of 3 kernels on the three examples in Figure 1. Note that for more complicated structures, i.e., examples b and c, DPK generates fewer fragments than SST and PTK, with more shared fragments among trees.

The complexity for all three kernels are at least $O(|N_1||N_2|)$ since they share the pairwise summation at the end of Equation 1. SST, due to its requirement of exact production rule matching, only takes one pass in the inner loop which adds a factor of ρ (the maximum branching factor of any production rule). DPK does a pairwise summation of children, which adds a factor of ρ^2 to the complexity. Finally, the efficient algorithm for PTK is proved by Moschitti (2006) to contain a constant factor of ρ^3 . Table 1 orders the tree kernels according to their listed complexity.

It may seem that the value of DPK is strictly in its ability to evaluate all paths, which is not explicitly accounted for by other kernels. However, another view of the DPK is possible by thinking of it as cheaply calculating rule production similarity by taking advantage of relatively strict English word ordering. Like SST and PTK, the DPK requires the root category of two subtrees to be the same for the similarity to be greater than zero. Unlike SST and PTK, once the root category comparison is successfully completed, DPK looks at all paths that go through it and accumulates their similarity scores independent of ordering – in other words, it will ignore the ordering of the children in its pro-

duction rule. This means, for example, that if the rule production $NP \rightarrow NN JJ DT$ were ever found in a tree, to DPK it would be indistinguishable from the common production $NP \rightarrow DT JJ NN$, despite having inverted word order, and thus would have a maximal similarity score. SST and PTK would assign this pair a much lower score for having completely different ordering, but we suggest that cases such as these are very rare due to the relatively strict word ordering of English. In most cases, the determiner of a noun phrase will be at the front, the nouns will be at the end, and the adjectives in the middle. So with small differences in production rules (one or two adjectives, extra nominal modifier, etc.) the PTK will capture similarity by comparing every possible partial rule completion, but the DPK can obtain higher and faster scores by just comparing one child at a time because the ordering is constrained by the language. This analysis does lead to a hypothesis for the general viability of the DPK, suggesting that in languages with freer word order it may give inflated scores to structures that are syntactically dissimilar if they have the same constituent components in different order.

Formally, Moschitti (2006) showed that SST is a special case of PTK when only the longest child sequence from each tree is considered. On the other end of the spectrum, DPK is a special case of PTK where the similarity between rules only considers child subsequences of length one.

4 Evaluation

We applied DPK to two published temporal relation extraction systems: (Miller et al., 2013) in the clinical domain and Cleartk-TimeML (Bethard, 2013) in the general domain respectively.

4.1 Narrative Container Discovery

The task here as described by Miller et al. (2013) is to identify the CONTAINS relation between a time expression and a same-sentence event from clinical notes in the THYME corpus, which has 78 notes of 26 patients. We obtained this corpus from the authors and followed their linear composite kernel setting:

$$K_C(s_1, s_2) = \tau \sum_{p=1}^P K_T(t_1^p, t_2^p) + K_F(f_1, f_2) \quad (2)$$

where s_i is an instance object composed of flat features f_i and a syntactic tree t_i . A syntactic tree t_i

can have multiple representations, as in Bag Tree (BT), Path-enclosed Tree (PET), and Path Tree (PT). For the tree kernel K_T , subset tree (SST) kernel was applied on each tree representation p . The final similarity score between two instances is the τ -weighted sum of the similarities of all representations, combined with the flat feature (FF) similarity as measured by a feature kernel K_F (linear or polynomial). Here we replaced the SST kernel with DPK and tested two feature combinations FF+PET and FF+BT+PET+PT. To fine tune parameters, we used grid search by testing on the default development data. Once the parameters were tuned, we tested the system performance on the testing data, which was set up by the original system split.

4.2 Cleartk-TimeML

We tested one sub-task from TempEval-2013 – the extraction of temporal relations between an event and time expression within the same sentence. We obtained the training corpus (TimeBank + AQUAINT) and testing data from the authors (Bethard, 2013). Since the original features didn't contain syntactic features, we created a PET tree extractor for this system. The kernel setting was similar to equation (2), while there was only one tree representation, PET tree, $P=1$. A linear kernel was used as K_F to evaluate the exact same flat features as used by the original system. We used the built-in cross validation to do grid search for tuning the parameters. The final system was tested on the testing data for reporting results.

4.3 Results and Discussion

Results are shown in Table 2. The top section shows THYME results. For these experiments, the DPK is superior when a syntactically-rich PET representation is used. Using the full feature set of Miller et al. (2013), SST is superior to DPK and obtains the best overall performance. The bottom section shows results on TempEval-2013 data, for which there is little benefit from either tree kernel.

Our experiments with THYME data show that DPK can capture something in the linguistically richer PET representation that the SST kernel cannot, but adding BT and PT representations decrease the DPK performance. As a shallow representation, BT does not have much in the way of descending paths for DPK to use. PT already ignores the production grammar by removing the inner tree nodes. DPK therefore cannot get useful information and may even get misleading cues from these two rep-

Features	K_T	P	R	F
THYME				
FF+PET	DPK	0.756	0.667	0.708
	SST	0.698	0.630	0.662
FF+BT+PET+PT	DPK	0.759	0.626	0.686
	SST	0.754	0.711	0.732
TempEval				
FF+PET	DPK	0.328	0.263	0.292
	SST	0.325	0.263	0.290
FF	-	0.309	0.266	0.286

Table 2: Comparison of tree kernel performance for temporal relation extraction on THYME and TempEval-2013 data.

resentations. These results show that, while DPK should not always replace SST, there are representations in which it is superior to existing methods. This suggests an approach in which tree representations are matched to different convolution kernels, for example by tuning on held-out data.

For TempEval-2013 data, adding syntactic features did not improve the performance significantly (comparing F-score of 0.290 with 0.286 in Table 3). Probably, syntactic information is not a strong feature for all types of temporal relations on TempEval-2013 data.

5 Conclusion

In this paper, we developed a novel convolution tree kernel (DPK) for measuring syntactic similarity. This kernel uses a descending path representation in trees to allow higher similarity scores on partially matching structures, while being simpler and faster than other methods for doing the same. Future work will explore 1) a composite kernel which uses DPK for PET trees, SST for BT and PT, and feature kernel for flat features, so that different tree kernels can work with their ideal syntactic representations; 2) incorporate dependency structures for tree kernel analysis 3) applying DPK to other relation extraction tasks on various corpora.

6 Acknowledgements

Thanks to Sean Finan for technically supporting the experiments. The project described was supported by R01LM010090 (THYME) from the National Library Of Medicine.

References

- Fabio Aioli, Giovanni Da San Martino, and Alessandro Sperduti. 2009. Route kernels for trees. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 17–24. ACM.
- Steven Bethard and James H Martin. 2007. Cu-tmp: temporal relation classification using syntactic and semantic features. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 129–132. Association for Computational Linguistics.
- Steven Bethard. 2013. Cleartk-timeml: A minimalist approach to TempEval 2013. In *Second Joint Conference on Lexical and Computational Semantics (*SEM)*, volume 2, pages 10–14.
- Nate Chambers. 2013. Navytime: Event and time ordering from raw text. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 73–77, Atlanta, Georgia, USA, June. Association for Computational Linguistics.
- Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Neural Information Processing Systems*.
- David Haussler. 1999. Convolution kernels on discrete structures. Technical report, University of California in Santa Cruz.
- Dirk Hovy, James Fan, Alfio Gliozzo, Siddharth Patwardhan, and Chris Welty. 2012. When did that happen?: linking events and relations to timestamps. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 185–193. Association for Computational Linguistics.
- Tetsuji Kuboyama, Kouichi Hirata, Hisashi Kashima, Kiyoko F Aoki-Kinoshita, and Hiroshi Yasuda. 2007. A spectrum tree kernel. *Information and Media Technologies*, 2(1):292–299.
- Hector Llorens, Estela Saquete, and Borja Navarro. 2010. Tipsem (english and spanish): Evaluating CRFs and semantic roles in TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 284–291. Association for Computational Linguistics.
- Timothy Miller, Steven Bethard, Dmitriy Dligach, Sameer Pradhan, Chen Lin, and Guergana Savova. 2013. Discovering temporal narrative containers in clinical text. In *Proceedings of the 2013 Workshop on Biomedical Natural Language Processing*, pages 18–26, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Seyed Abolghasem Mirroshandel, M Khayyamian, and GR Ghassem-Sani. 2009. Using tree kernels for classifying temporal relations between events. *Proc. of the PACLIC23*, pages 355–364.
- Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Machine Learning: ECML 2006*, pages 318–329. Springer.
- James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. 2003. The TimeBank corpus. In *Corpus linguistics*, volume 2003, page 40.
- Geoffrey Sampson. 2000. A proposal for improving the measurement of parse accuracy. *International Journal of Corpus Linguistics*, 5(1):53–68.
- Stephanie Strassel, Dan Adams, Henry Goldberg, Jonathan Herr, Ron Keesing, Daniel Oblinger, Heather Simpson, Robert Schrag, and Jonathan Wright. 2010. The DARPA machine reading program-encouraging linguistic and reasoning research with a series of reading tasks. In *LREC*.
- William Styler, Steven Bethard, Sean Finan, Martha Palmer, Sameer Pradhan, Piet de Groen, Brad Erickson, Timothy Miller, Lin Chen, Guergana K. Savova, and James Pustejovsky. 2014. Temporal annotations in the clinical domain. *Transactions of the Association for Computational Linguistics*, 2(2):143–154.
- Min Zhang, Wanxiang Che, Ai Ti Aw, Chew Lim Tan, Guodong Zhou, Ting Liu, and Sheng Li. 2007. A grammar-driven convolution tree kernel for semantic role classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 200–207.

Entities' Sentiment Relevance

Zvi Ben-Ami

The Hebrew University
Jerusalem, ISRAEL

`zvi.benami@mail.huji.ac.il`

Ronen Feldman

The Hebrew University
Jerusalem, ISRAEL

`ronen.feldman@huji.ac.il`

Binyamin Rosenfeld

Digital Trowel
New York, USA

`grurgrur@gmail.com`

Abstract

Sentiment relevance detection problems occur when there is a sentiment expression in a text, and there is the question of whether or not the expression is related to a given entity or, more generally, to a given situation. The paper discusses variants of the problem, and shows that it is distinct from other somewhat similar problems occurring in the field of sentiment analysis and opinion mining. We experimentally demonstrate that using the information about relevancy significantly affects the final sentiment evaluation of the entities. We then compare a set of different algorithms for solving the relevance detection problem. The most accurate results are achieved by algorithms that use certain document-level information about the target entities. We show that this information can be accurately extracted using supervised classification methods.

1 Introduction

Sentiment extraction by modern sentiment analysis (SA) systems is usually based on searching the input text for sentiment-bearing words and expressions, either general (language-wide) or domain-specific. In most common SA approaches, each such expression carries a polarity value ("positive" or "negative") which is possibly weighted. The sum of all polarity values from all expressions found in a text becomes the sentiment score for the whole text.

People are, however, usually interested in sentiments regarding some entity or situation, and not in sentiments of a particular document. A natural way to make the SA more focused is to explicitly bind each sentiment expression to a specific entity, or to a small set of entities from among all entities mentioned in the document.

The choice of which entity to bind a sentiment expression to, can be made according to the proximity (physical, syntactical, and/or semantic) and/or salience of the entities.

In this paper, we argue that all of these methods can be useful in different contexts, and so the best single algorithm should use all available proximity information, of all kinds, together with additional context information – position in the document, section, or paragraph; proximity of other entities; lexical contents; etc. One of the most important context information is the type of relation between the target entity and the document – whether the entity is the main topic of the document, or one of several main topics, or mentioned in passing, etc.

Another layer that we'd like to add concerns the interaction of different entity types during SA. In a typical situation, there is only one entity type which is the target for SA. In such cases, clearly distinguishing between the relevancy of target and non-target entities types is not essential. For example, when the general topic is a COMPANY, and there is a sentiment expression referring to a PERSON or a PRODUCT, this sentiment expression is still relevant to the company and can be regarded as such. In other situations, SA users may be specifically interested in an interaction between entities of different types. For example, in a medical forum setting, it may be interesting to know the users' sentiments regarding a given DRUG in the context of a given DISEASE. We will show that such situations are modeled well enough using intersections of regions of relevance of the participating entity types, with the relevance region for each type calculated separately.

We purposefully exclude possible interactions between entities of the same type, because they behave in a different way. The precise analysis of such interactions is a different topic from rele-

vance detection, and so it is mostly ignored in this paper.

2 Related Work

The task of SA has drawn the attention of many researchers worldwide (Connor et al., 2010; Liu, 2012; Loughran and McDonald, 2010; Pang and Lee, 2004; Turney, 2002). While most SA research is focused on discovering and classifying the expressions, some are also concerned with the targets of the expressions and explicitly identify the syntactic targets of sentiment expressions (Pang and Lee, 2004).

Other related works belong to the Passage Retrieval field, since the relevance detection problem can be construed as a specific form of passage retrieval problem (Liu and Croft, 2002; Tiedemann and Mur, 2008). Different approaches were suggested for passage retrieval (Buscaldi et al., 2010; Comas et al., 2012; Hearst, 1997; Lafferty et al., 2001; Lin et al., 2012; Liu and Croft, 2002; Lloret et al., 2012; O'Connor et al., 2013; Otterbacher et al., 2009; Salton et al., 1993; Wachsmuth, 2013), some are more sophisticated than others.

The closest approach to ours is the one of Scheible and Schütze (2013), but in contrast to them, we strive to discover sentiments' relevance for all entities (of a given type) mentioned in the document, not necessarily topical.

3 Entity Relevance

An instance of the sentiment relevance detection problem for a single entity consists of a text document, a sentiment expression within the document, and a target entity. The task is a binary decision: 'relevant' vs. 'irrelevant'. To solve this task, we can use any information that can be found by analyzing the document. Thus, we can assume that we know the parse trees of all sentences and the locations of all references of all entities in the document, including co-references.

In addition, we make use of an extra piece of information for each target entity – its "status within the document", or "document type with respect to the entity". We distinguish between several types which are intuitively clearly different:

- **'Target'** – the entity is the main topic of the document;
- **'Accidental'** – the entity is not the main topic of the document, and is mentioned in passing;

- **'RelationTarget'** – the main topic of the document is a relation between the entity and some other entities of the same type;
- **'ListTarget'** – the entity is one of a few equally important topics, dealt with sequentially.

In the datasets we use for experiments, each entity is manually annotated with its status within the document, which allows us to directly observe the influence of this data on the accuracy of relevance discernment. We also show that this data can be automatically extracted using supervised classification.

Since this paper is primarily a study of sentiment relevance, the actual sentiment expressions are not always labeled in our datasets. Instead, relevance ranges are annotated for each entity, in the style of passage retrieval problems, with the expectation that sentiment expressions relevant to an entity only appear in the parts of the document that are labeled as "relevant", and conversely, that all expressions appearing in parts labeled "irrelevant" are irrelevant. This way of annotating allows the comparing of different relevance detection strategies independently of the main sentiment extraction tool.

All of the algorithms discussed in this paper use the same document processing methods, thus allowing us to compare the algorithms themselves independent of the quality and specifics of the underlying NLP.

The multiple-entity relevance problem is distinguished from the single-entity relevance problem by the requirement for the sentiment expression to be relevant to several entities of different types. The problem is close to Relation Extraction in this sense. The examples we are interested in are in the medical domain and deal with three main entity types: PERSON, DRUG, and DISEASE, where PERSON is restricted to known physicians. While each of the entity types can be the target of a sentiment expression, the more interesting questions in this domain involve multiple entities, specifically, DRUG + DISEASE ("how effective is this drug for this disease?"), and PERSON + DRUG + DISEASE ("what does this physician say about using this drug to cure this disease?").

We solve the multiple-entity relevance problem by intersecting the relevance ranges of different-type entities, thus reducing the problem to the single-entity relevance detection. As such, the experiments regarding the multiple-entity relevance need only check the accuracy of this reduction. In the medical domain, at least, this accuracy appears to be adequate.

4 Relevance Algorithms

Each algorithm receives, as input, the text of the document, with labeled reference of the target entity and other entities of the same type. The labeled references also include all coreferential references, extracted automatically by an NLP system. The input text also includes labeled candidate sentiment expressions, either manually labeled or automatically extracted by a relevance-ignoring SA system¹. The task of the algorithms is to label each candidate expression as relevant or irrelevant to the target entity. The algorithms are evaluated according to the accuracy (recall, precision, and F1) of this labeling of individual sentiment expressions.

This method produces a reasonably well-understandable quality measure (the percentage of expressions that the algorithms get right or wrong), and also allows us to compare algorithms focused on individual expressions and algorithms working on text ranges. The algorithms we evaluate are as follows:

- **Baseline** - Every expression is declared relevant. This is the standard mode of operation of document-level SA tools, although it is usually only applied to the 'Target' entities – the main topic(s) of the document.
- **Physical-proximity-based** - A text-range focused algorithm, which labels pieces of text as relevant or irrelevant according to their placement relative to the references of the target entity and other entities of the same type, as well as some other contextual clues, such as paragraph boundaries. Generally, the mentioning of an entity starts its relevance range (and stops the relevance range of the previously mentioned entity). For the first entity reference in a paragraph, the range also extends backward to the beginning of the sentence. There are three flavors of the algorithm, specifically adapted for different document-types-with-respect-to-the-target-entity:
 - **'Proximity-Accidental'** - stops relevance ranges at paragraph boundaries,
 - **'Proximity-Targeted'** - restarts relevance ranges at paragraph boundaries (every para-

graph is assumed relevant at the start, unless another entity is mentioned).

- **'Proximity-List'** - interpolates relevance ranges over intermission paragraphs, unless they are explicitly irrelevant (e.g., containing references of other entities of the same type).
- **Syntactic-proximity-based** - An expression-focused algorithm, which labels expressions as relevant or irrelevant according to their distance to various entity references in the dependency parse graph. There are two flavors of the algorithm: direct and reverse. The former considers an expression relevant only if it is closest to the target entity from among all entities of the same type, and the distance is sufficiently close. The latter considers an expression irrelevant only if it has the above-described relation to some non-target entity of the same type. The rationale for the two flavors is the distinction between 'Targeted' and 'Accidental' document types regarding the target entity. For the 'Accidental' entities, a sentiment expression is assumed to be relevant only if it is explicitly connected to the entity. For 'Targeted' entities, an expression is irrelevant only if it is explicitly connected to some other entity of the same type.
- **Classification-based** - This algorithm considers each candidate sentiment expression as an instance of a binary classification problem, to be solved using supervised classification. For evaluating this algorithm, some part of the test corpus is used for training, and the other for testing, with N-fold cross-validation. The features for classification may use any information present in the input.

In the current experiments, we use references of target and non-target entities, appearances of paragraph and document boundaries, length of syntactic connections to target and non-target entities, when available, and explicit entity status within documents, when available. The (binary) classification features are built from sequences of up to 5 occurrences of the above-described pieces, with the pieces appearing before and after the sentiment expression tracked separately. For classification, we use a linear classifier with Large Margin training (regularized perceptron, as discussed in Scheible and Schütze, (2013)).
- **Sequence-classification-based** - The algorithm uses exactly the same features as the direct classification-based above, but instead of considering each expression separately, it con-

¹In our experiments, we also use a standalone automatic Financial SA system from Feldman et al. (2010), working in the 'ignore relevance' mode, which (1) finds and labels all entities of the target type(s); (2) resolves all coreferences for the target entity type(s); (3) finds and labels all sentiment expressions, regardless of their relevance; and (4) provides dependency parses for all sentences in the corpus.

siders them as a sequence, one per document. So, instead of a Large Margin binary classifier, a probabilistic sequence classifier is used (CRF, as discussed in Lafferty et al. (2001)).

5 Experiments

For the experiments, we use two manually-annotated corpora², a financial corpus³ and a medical⁴ corpus. In the Financial corpus, COMPANIES are used as target entities and in the medical corpus, DISEASEs, DRUGs and PERSONs are the entity types that are used as target entities. For the purpose of the experiments, we are interested only in single-entity sentiments about DRUGs, and multiple-entity sentiments about DRUGs + DISEASEs, and DRUGs + DISEASEs + PERSONs.

The evaluation metrics in all of the experiments are precision, recall, and F1. For the classification-based algorithms, unless stated otherwise, we use 10-fold cross-validation.

5.1 Experiment: Importance of relevance

In the first experiment, we demonstrate the importance of using relevance when calculating the consolidated sentiment score of an entity within a set of documents. For each entity, we set the 'correct' consolidated sentiment score to the average of polarities of all sentiments in a corpus which are labeled as relevant to the entity. Then, we compare the correct value to the two scores calculated without considering relevance:

- **'Baseline'** - the average of polarities of all sentiments in all documents where the entity is mentioned, and
- **'TargetedOnly'** - the average of polarities of all sentiments in the documents where the entity is labeled as target (main topic of the document). This case models the typical state of a relevance-agnostic SA system.

For this evaluation, we only compare the sign of the final sentiment scores, without considering their magnitudes (unless it is close to zero, in

which it is considered 'neutral'). The errors at this level indicate definite SA errors – miscalculating entity's sentiment into its opposite.

The results of the evaluation are as follows: The 'Baseline' scores show a large difference from the correct scores, with 33% and 38% of entities having wrong final polarity in the financial (COMPANY) and medical (DRUG) domains, respectively. The 'TargetedOnly' scores are somewhat closer to correct, with 12% and 28% of entities with incorrect final polarities. However, the 'TargetedOnly' method naturally suffers from a very low recall, with only 19% and 38% of entities covered in the financial and medical domains, respectively.

5.2 Experiment: Influence of entity status

In this experiment, we compare the performance of various algorithms while either providing or withholding the information about the document-type-with-respect-to-the-target-entity.

The performance of the physical proximity algorithms on the financial corpus is shown at the top left hand side of Table 1. The set of all instances of relevance detection problems in the corpus (an instance consists of a sentiment expression within a text, together with a target entity) is divided into three subsets, according to the status of the target entity within the document. As expected, the three flavors of the physical proximity algorithm perform much better on the corpus subsets they are adapted to. At the bottom left hand side of Table 1, we similarly show the performance of the two flavors of the syntax-proximity-based algorithm on the medical domain (DRUG entities). Same as above, there is a large difference in the performance of the two flavors of the algorithm on different subsets of the problem set. Finally, at the top of Table 2, we compare the performance of the two classification-based algorithms on the two (whole) problem sets, while either keeping or withholding the entity status information from the classifier. The difference in results is less pronounced here, but is still noticeable. The reason for the smaller difference, we hypothesize, is the ability of the classifiers to partially infer the entity status from the various context clues that are used as classification features (see the experiment 5.3).

5.3 Experiment: Automatic identification of entity status using classification.

In this experiment, we confirm that it is possible to identify the entity status within documents using supervised classification.

² Fully annotating texts for semantic relevance is an arduous task, thus the used annotated corpora are relatively small. Sample can be found at <http://goo.gl/6HONHP>.

³ A corpus of 160 financial news documents on at least one entity of interest, of average size ~5Kb, downloaded from various financial news websites. The dataset mentions 424 different companies.

⁴ A corpus of 160 documents, of average size ~7Kb, downloaded following Google queries on a set of a few common drugs and diseases. The dataset mentions 722 different people, 46 diseases, and 175 drugs.

	Experiment 5.2 (Precision/Recall/F1)				Experiment 5.3 (F1, (diff. in F1 from exp. 5.2))			
	Accidental	Targeted	List	Whole	Accidental	Targeted	List	Whole
Proximity-Accidental	84/43/ 57	93/76/84	92/74/82	92/72/81	60 (+2.6)	79 (-5.5)	83 (+1.1)	
Proximity-Targeted	31/50/38	90/ 84 /87	55/89/68	63/83/72	38 (-0.4)	82 (-5.2)	73 (+4.3)	
Proximity-List	58/44/50	90/83/87	88 /83/ 86	85/80/82	52 (+2.1)	81 (-5.9)	87 (+1.6)	
Proximity-Combined				89/80/84				83 (-1.2)
Syntactic-Prox.-Direct	93/48/ 64	99/42/60			65 (+0.8)	59 (-0.2)		
Syntactic-Prox.-Inverse	04/72/08	70/66/ 68			8 (-0.2)	76 (+6.4)		

Table 1. Performance of different algorithms on three subsets of the corpus with a different status of the target entity within the document.

Experiment	Algorithm	Financial	Medical
Experiment 5.2 (Prec./ Rec./F1).	Classification (with entity status info)	90/86/ 88	84/88/ 86
	Classification (without entity status info)	89/85/87	87/81/84
	Sequence Classification (with entity status info)	96/84/ 90	99/84/ 91
	Sequence Classification (without entity status info)	96/83/89	95/85/90
Experiment 5.3 (F1, (diff. in F1 from exp. 5.2))	Classification	86.7 (-0.9)	83.9 (-2.0)
	Sequence Classification	89.7 (+0.1)	90.9 (-0.3)
Experiment 5.5 (F1)	Baseline	37.2	28.6
	Physical Proximity	84.1	79.5
	Syntactic-Proximity	43.8	54.6
	Classification	87.6	85.9
	Sequence-Classification	91.2	89.6

Table 2. Performance of different algorithms on the different domains.

The results of direct evaluation show that the accuracies of the Medical and Financial corpora (using 10-fold X-validation) are 87.8% and 82.2% respectively, and the accuracy when using the Medical corpus for training the Financial corpus for testing and vice versa, are 78.2% and 86.1% , respectively.

The results of relevance detection using the automatically extracted entity status values are shown at the right hand side of Table 1 and in the middle of Table 2, which utilize the same datasets and algorithms as at the left hand side of Table 1 and at the top of Table 2. As can be seen from the tables, the drop in performance is small, demonstrating the success of classification-based extraction of entity status information.

5.4 Experiment: Cross-domain applicability

In this experiment, we test how well the classifiers trained on data from one domain work on input from a different domain.

The classification results using different types of training data are shown in Table 3.

	Classification	Sequence classification
Medical 2-fold/10-fold	84.6/85.9	85.7/89.6
Train on Fin, test on Med	83.5	86.8
Financial 2-fold/10-fold	86.1/87.6	90.3/91.2
Train on Med, test on Fin	85.4	91.0

Table 3. Performance of classification-based algorithms using different training data (F₁).

The table confirms general independence of the classification performance on the domain. Comparing the 2-fold and 10-fold cross-validation results (the difference is equivalent to doubling the amount of training data), shows that the amount of training data is sufficient.

5.5 Experiment: Overall performance of algorithms

In this experiment, we simply compare the overall accuracy of various algorithms for relevance discernment, operating at their best parameters. The results are shown at the bottom of Table 2. Overall, classification-based algorithms perform better than the deterministic ones, with sequence-classification performing significantly better than direct classification. Syntactic proximity-based is precise, but has relatively low recall, reducing its overall performance. Physical proximity-based is simplest, and produce reasonably high overall results, although worse than the best-performing classification-based methods.

6 Conclusion

The results are mostly intuitively understood and confirm the expectations. We confirmed that relevance detection is essential for producing correct consolidated SA results. We found that the entity status within the document is one of the important clues for solving the relevance detection problem, and showed that this information can be effectively automatically extracted using supervised classification. We also compared several algorithms for relevance detection, with the results that classification-based algorithms generally outperform simpler ones based on the same clues, although a very simple proximity-based algorithm performs reasonably well if allowed to use the entity status information.

Acknowledgments

This work is supported by the Israel Ministry of Science and Technology Center of Knowledge in Machine Learning and Artificial Intelligence and the Israel Ministry of Defense.

References

- Buscaldi, D., Rosso, P., Gómez-Soriano, J., Sanchis, E., 2010. Answering questions with an n-gram based passage retrieval engine. *J. Intell. Inf. Syst.* 34, 113–134. doi:10.1007/s10844-009-0082-y
- Comas, P.R., Turmo, J., Màrquez, L., 2012. Sibyl, a factoid question-answering system for spoken documents. *ACM Trans. Inf. Syst.* 30, 19:1–19:40. doi:10.1145/2328967.2328972
- Connor, B.O., Balasubramanyan, R., Routledge, B.R., Smith, N.A., 2010. From Tweets to Polls : Linking Text Sentiment to Public Opinion Time Series, in: *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media.* pp. 122–129.
- Feldman, R., Rosenfeld, B., Bar-haim, R., Fresko, M., 2010. The Stock Sonar — Sentiment Analysis of Stocks Based on a Hybrid Approach, in: *Proceedings of the Twenty-Third Innovative Applications of Artificial Intelligence Conference.* pp. 1642–1647.
- Hearst, M.A., 1997. TextTiling: segmenting text into multi-paragraph subtopic passages. *Comput. Linguist.* 23, 33–64.
- Lafferty, J., McCallum, A., Pereira, F., 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data., in: *Proceedings of the Eighteenth International Conference on Machine Learning (ICML-2001).*
- Lin, H.-T., Chi, N.-W., Hsieh, S.-H., 2012. A concept-based information retrieval approach for engineering domain-specific technical documents. *Adv. Eng. Informatics* 26, 349–360. doi:http://dx.doi.org/10.1016/j.aei.2011.12.003
- Liu, B., 2012. *Sentiment Analysis and Opinion Mining Synthesis Lectures on Human Language Technologies.* Morgan & Claypool Publishers.
- Liu, X., Croft, W.B., 2002. Passage retrieval based on language models, in: *Proceedings of the Eleventh International Conference on Information and Knowledge Management, CIKM '02.* ACM, New York, NY, USA, pp. 375–382. doi:10.1145/584792.584854
- Lloret, E., Balahur, A., Gómez, J., Montoyo, A., Palomar, M., 2012. Towards a unified framework for opinion retrieval, mining and summarization. *J. Intell. Inf. Syst.* 39, 711–747. doi:10.1007/s10844-012-0209-4
- Loughran, T.I.M., McDonald, B., 2010. When is a Liability not a Liability? Textual Analysis , Dictionaries , and 10-Ks Journal of Finance , forthcoming. *J. Finance* 66, 35–65.
- O'Connor, B., Stewart, B.M., Smith, N.A., 2013. Learning to Extract International Relations from Political Context, in: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* Association for Computational Linguistics, Sofia, Bulgaria, pp. 1094–1104.
- Otterbacher, J., Erkan, G., Radev, D.R., 2009. Biased LexRank: Passage retrieval using random walks with question-based priors. *Inf. Process. Manag.* 45, 42–54. doi:http://dx.doi.org/10.1016/j.ipm.2008.06.004
- Pang, B., Lee, L., 2004. A Sentimental Education : Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts.
- Salton, G., Allan, J., Buckley, C., 1993. Approaches to passage retrieval in full text information systems, in: *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '93.* ACM, New York, NY, USA, pp. 49–58. doi:10.1145/160688.160693
- Scheible, C., Schütze, H., 2013. Sentiment Relevance, in: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* Association for Computational Linguistics, Sofia, Bulgaria, pp. 954–963.
- Tiedemann, J., Mur, J., 2008. Simple is best: experiments with different document segmentation strategies for passage retrieval, in: *Coling 2008: Proceedings of the 2nd Workshop on Information Retrieval for Question Answering, IRQA '08.* Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 17–25.
- Turney, P., 2002. Thumbs Up or Thumbs Down ? Semantic Orientation Applied to Unsupervised Classification of Reviews, in: *Proceedings of the Association for Computational Linguistics (ACL).* pp. 417–424.
- Wachsmuth, H., 2013. Information Extraction as a Filtering Task Categories and Subject Descriptors, in: *To Appear in Proc. of the 22th ACM CIKM.*

Automatic Detection of Multilingual Dictionaries on the Web

Gintarė Grigonytė[♠] Timothy Baldwin[♡]

[♠] Department of Linguistics, Stockholm University

[♡] Department of Computing and Information Systems, The University of Melbourne

gintare@ling.su.se tb@ldwin.net

Abstract

This paper presents an approach to query construction to detect multilingual dictionaries for predetermined language combinations on the web, based on the identification of terms which are likely to occur in bilingual dictionaries but not in general web documents. We use eight target languages for our case study, and train our method on pre-identified multilingual dictionaries and the Wikipedia dump for each of our languages.

1 Motivation

Translation dictionaries and other multilingual lexical resources are valuable in a myriad of contexts, from language preservation (Thieberger and Berez, 2012) to language learning (Laufer and Hadar, 1997), cross-language information retrieval (Nie, 2010) and machine translation (Munteanu and Marcu, 2005; Soderland et al., 2009). While there are syndicated efforts to produce multilingual dictionaries for different pairings of the world’s languages such as `freedict.org`, more commonly, multilingual dictionaries are developed in isolation for a specific set of languages, with ad hoc formatting, great variability in lexical coverage, and no central indexing of the content or existence of that dictionary (Baldwin et al., 2010). Projects such as `panlex.org` aspire to aggregate these dictionaries into a single lexical database, but are hampered by the need to identify individual multilingual dictionaries, especially for language pairs where there is a sparsity of data from existing dictionaries (Baldwin et al., 2010; Kamholz and Pool, to appear). This paper is an attempt to automate the detection of multilingual dictionaries on the web, through query construction for an arbitrary language pair. Note that for the method to work,

we require that the dictionary occurs in “list form”, that is it takes the form of a single document (or at least, a significant number of dictionary entries on a single page), and is not split across multiple small-scale sub-documents.

2 Related Work

This research seeks to identify documents of a particular type on the web, namely multilingual dictionaries. Related work broadly falls into four categories: (1) mining of parallel corpora; (2) automatic construction of bilingual dictionaries/thesauri; (3) automatic detection of multilingual documents; and (4) classification of document genre.

Parallel corpus construction is the task of automatically detecting document sets that contain the same content in different languages, commonly based on a combination of site-structural and content-based features (Chen and Nie, 2000; Resnik and Smith, 2003). Such methods could potentially identify parallel word lists from which to construct a bilingual dictionary, although more realistically, bilingual dictionaries exist as single documents and are not well suited to this style of analysis.

Methods have also been proposed to automatically construct bilingual dictionaries or thesauri, e.g. based on crosslingual glossing in predictable patterns such as a technical term being immediately preceded by that term in a lingua franca source language such as English (Nagata et al., 2001; Yu and Tsujii, 2009). Alternatively, comparable or parallel corpora can be used to extract bilingual dictionaries based on crosslingual distributional similarity (Melamed, 1996; Fung, 1998). While the precision of these methods is generally relatively high, the recall is often very low, as there is a strong bias towards novel technical terms being glossed but more conventional terms not.

Also relevant to this work is research on lan-

guage identification, and specifically the detection of multilingual documents (Prager, 1999; Yamaguchi and Tanaka-Ishii, 2012; Lui et al., 2014). Here, multi-label document classification methods have been adapted to identify what mix of languages is present in a given document, which could be used as a pre-filter to locate documents containing a given mixture of languages, although there is, of course, no guarantee that a multilingual document is a dictionary.

Finally, document genre classification is relevant in that it is theoretically possible to develop a document categorisation method which classifies documents as multilingual dictionaries or not, with the obvious downside that it would need to be applied exhaustively to all documents on the web. The general assumption in genre classification is that the type of a document should be judged not by its content but rather by its form. A variety of document genre methods have been proposed, generally based on a mixture of structural and content-based features (Matsuda and Fukushima, 1999; Finn et al., 2002; zu Eissen and Stein, 2005).

While all of these lines of research are relevant to this work, as far as we are aware, there has not been work which has proposed a direct method for identifying pre-existing multilingual dictionaries in document collections.

3 Methodology

Our method is based on a query formulation approach, and querying against a pre-existing index of a document collection (e.g. the web) via an information retrieval system.

The first intuition underlying our approach is that certain words are a priori more “language-discriminating” than others, and should be preferred in query construction (e.g. *sushi* occurs as a [transliterated] word in a wide variety of languages, whereas *anti-discriminatory* is found predominantly in English documents). As such, we prefer search terms w_i with a higher value for $\max_l P(l|w_i)$, where l is the language of interest.

The second intuition is that the lexical coverage of dictionaries varies considerably, especially with multilingual lexicons, which are often compiled by a single developer or small community of developers, with little systematicity in what is including or not included in the dictionary. As such, if we are to follow a query construction approach to lexicon discovery, we need to be able

to predict the likelihood of a given word w_i being included in an arbitrarily-selected dictionary D_l incorporating language l (i.e. $P(w_i|D_l)$). Factors which impact on this include the lexical prior of the word in the language (e.g. $P(\textit{paper}|\textit{en}) > P(\textit{papyrus}|\textit{en})$), whether they are lemmas or not (noting that multilingual dictionaries tend not to contain inflected word forms), and their word class (e.g. multilingual dictionaries tend to contain more nouns and verbs than function words).

The third intuition is that certain word *combinations* are more selective of multilingual dictionaries than others, i.e. if certain words are found together (e.g. *cruiser*, *gospel* and *noodle*), the containing document is highly likely to be a dictionary of some description rather than a “conventional” document.

Below, we describe our methodology for query construction based on these elements in greater detail. The only assumption on the method is that we have access to a selection of dictionaries D (mono- or multilingual) and a corpus of conventional (non-dictionary) documents C , and knowledge of the language(s) contained in each dictionary and document.

Given a set of dictionaries D_l for a language l and the complement set $D_{\bar{l}} = D \setminus D_l$, we first construct the lexicon L_l for that language as follows:

$$L_l = \{w_i | w_i \in D_l \cap w_i \notin D_{\bar{l}}\} \quad (1)$$

This creates a language-discriminating lexicon for each language, satisfying the first criterion.

Lexical resources differ in size, scope and coverage. For instance, a well-developed, mature multilingual dictionary may contain over 100,000 multilingual lexical records, while a specialised 5-way multilingual domain dictionary may contain as few as 100 multilingual lexical records. In line with our second criterion, we want to select words which have a higher likelihood of occurrence in a multilingual dictionary involving that language. To this end, we calculate the weight $\text{sdict}(w_{i,l})$ for each word $w_{i,l} \in L_l$:

$$\text{sdict}(w_{i,l}) = \sum_{d \in D_l} \begin{cases} \frac{|L_l| - |d|}{|L_l|} & \text{if } w_{i,l} \in d \\ -\frac{|d|}{|L_l|} & \text{otherwise} \end{cases} \quad (2)$$

where $|d|$ is the size of dictionary d in terms of the number of lexemes it contains.

The final step is to weight words by their typicality in a given language, as calculated by their

likelihood of occurrence in a random document in that language. This is estimated by the proportion of Wikipedia documents in that language which contain the word in question:

$$\text{Score}(w_{i,l}) = \frac{df(w_{i,l})}{N_l} \text{sdict}(w_{i,l}) \quad (3)$$

where $df(w_{i,l})$ is the count of Wikipedia documents of language l which contain w_i , and N_l is the total number of Wikipedia documents in language l .

In all experiments in this paper, we assume that we have access to at least one multilingual dictionary containing each of our target languages, but in absence of such a dictionary, $\text{sdict}(w_{i,l})$ could be set to 1 for all words $w_{i,l}$ in the language.

The result of this term weighing is a ranked list of words for each language. The next step is to identify combinations of words that are likely to be found in multilingual dictionaries and not standard documents for a given language, in accordance with our third criterion.

3.1 Apriori-based query generation

We perform query construction for each language based on frequent item set mining, using the Apriori algorithm (Agrawal et al., 1993). For a given combination of languages (e.g. English and Swahili), queries are then formed simply by combining monolingual queries for the component languages.

The basic approach is to use a modified support formulation within the Apriori algorithm to prefer word combinations that do not cooccur in regular documents. Based on the assumption that querying a (pre-indexed) document collection is relatively simple, we generate a range of queries of decreasing length and increasing likelihood of term co-occurrence in standard documents, and query until a non-empty set of results is returned.

The modified support formulation is as follows:

$$\text{cscore}(w_1, \dots, w_n) = \begin{cases} 0 & \text{if } \exists d, w_i, w_j : \text{co}_d(w_i, w_j) \\ \prod_i \text{Score}(w_i) & \text{otherwise} \end{cases}$$

where $\text{co}_d(w_i, w_j)$ is a Boolean function which evaluates to true iff w_i and w_j co-occur in document d . That is, we reject any combinations of words which are found to co-occur in Wikipedia documents for that language. Note that the actual calculation of this co-occurrence can be performed

en - natives unenjoyable
de - andeuten tau anwuchs fÜgung
fr - collègue étouffée hybride
es - encendedor juntarse tensión
it - ardenne gradevole calcolare mancia
ar - الجيب أقواس الحربية الصانع الشمال
zh - 球员 胡同 粒子
ja - 冷房 メモリ 巡洋艦 福音 井

Figure 1: Examples of learned queries for different languages

efficiently, as: (a) for a given iteration of Apriori, it only needs to be performed between the new word that we are adding to the query (“item set” in the terminology of Apriori) and each of the other words in a non-zero support itemset from the previous iteration of the algorithm (which are guaranteed to not co-occur with each other); and (b) the determination of whether two terms collocate can be performed efficiently using an inverted index of Wikipedia for that language.

In our experiments, we apply the Apriori algorithm exhaustively for a given language with a support threshold of 0.5, and return the resultant item sets in ranked order of combined score for the component words.

A random selection of queries learned for each of the 8 languages targeted in this research is presented in Figure 1.

4 Experimental methodology

We evaluate our proposed methodology in two ways:

1. against a synthetic dataset, whereby we injected bilingual dictionaries into a collection of web documents, and evaluated the ability of the method to return multilingual dictionaries for individual languages; in this, we naively assume that all web documents in the background collection are not multilingual dictionaries, and as such, the results are potentially an underestimate of the true retrieval effectiveness.
2. against the open web via the Google search API for a given combination of languages, and hand evaluation of the returned documents

Lang	Wikipedia articles (M)	Dictionaries	Queries learned	Avg. query length
en	3.1	26	2546	3.2
zh	0.3	0	5034	3.6
es	0.5	2	356	2.9
ja	0.6	0	1532	3.3
de	1.0	13	634	2.7
fr	0.9	5	4126	3.0
it	0.6	4	1955	3.0
ar	0.1	2	9004	3.2

Table 1: Details of the training data and queries learned for each language

Note that the first evaluation with the synthetic dataset is based on *monolingual* dictionary retrieval effectiveness because we have very few (and often no) multilingual dictionaries for a given pairing of our target languages. For a given language, we are thus evaluating the ability of our method to retrieve multilingual dictionaries containing that language (and other indeterminate languages).

For both the synthetic dataset and open web experiments, we evaluate our method based on mean average precision (MAP), that is the mean of the average precision scores for each query which returns a non-empty result set.

To train our method, we use 52 bilingual Freedict (Freedict, 2011) dictionaries and Wikipedia¹ documents for each of our target languages. As there are no bilingual dictionaries in Freedict for Chinese and Japanese, the training of Score values is based on the Wikipedia documents only. Morphological segmentation for these two languages was carried out using MeCab (MeCab, 2011) and the Stanford Word Segmenter (Tseng et al., 2005), respectively. See Table 1 for details of the number of Wikipedia articles and dictionaries for each language.

Below, we detail the construction of the synthetic dataset.

4.1 Synthetic dataset

The synthetic dataset was constructed using a subset of ClueWeb09 (ClueWeb09, 2009) as the background web document collection. The original ClueWeb09 dataset consists of around 1 billion web pages in ten languages that were collected in January and February 2009. The relative proportions of documents in the different languages in the original dataset are as detailed in Table 2.

We randomly downsampled ClueWeb09 to 10

¹Based on 2009 dumps.

Language	Proportion
en (English)	48.41%
zh (Chinese)	17.05%
es (Spanish)	7.62%
ja (Japanese)	6.47%
de (German)	4.89%
fr (French)	4.79%
ko (Korean)	3.61%
it (Italian)	2.8%
pt (Portuguese)	2.62%
ar (Arabic)	1.74%

Table 2: Language proportions in ClueWeb09.

million documents for the 8 languages targeted in this research (the original 10 ClueWeb09 languages minus Korean and Portuguese). We then sourced a random set of 246 multilingual dictionaries that were used in the construction of `panlex.org`, and injected them into the document collection. Each of these dictionaries contains at least one of our 8 target languages, with the second language potentially being outside the 8. A total of 49 languages are contained in the dictionaries.

We indexed the synthetic dataset using Indri (Indri, 2009).

5 Results

First, we present results over the synthetic dataset in Table 3. As our baseline, we simply query for the language name and the term *dictionary* in the local language (e.g. *English dictionary*, for English) in the given language.

For languages that had bilingual dictionaries for training, the best results were obtained for Spanish, German, Italian and Arabic. Encouragingly, the results for languages with only Wikipedia documents (and no dictionaries) were largely comparable to those for languages with dictionaries, with Japanese achieving a MAP score comparable to the best results for languages with dictionary training data. The comparably low result for

Lang	Dicts	MAP	Baseline
en	92	0.77	0.00
zh	7	0.75	0.00
es	34	0.98	0.04
ja	5	0.94	0.00
de	75	0.97	0.08
fr	34	0.84	0.03
it	8	0.95	0.01
ar	3	0.92	0.00
AVERAGE:	32.2	0.88	0.04

Table 3: Dictionary retrieval results over the synthetic dataset (“Dicts” = the number of dictionaries in the document collection for that language).

English is potentially affected by its prevalence both in the bilingual dictionaries in training (restricting the effective vocabulary size due to our L_l filtering), and in the document collection. Recall also that our MAP scores are an underestimate of the true results, and some of the ClueWeb09 documents returned for our queries are potentially relevant documents (i.e. multilingual dictionaries including the language of interest). For all languages, the baseline results were below 0.1, and substantially lower than the results for our method.

Looking next to the open web, we present in Table 4 results based on querying the Google search API with the 1000 longest queries for English paired with each of the other 7 target languages. Most queries returned no results; indeed, for the en-ar language pair, only 49/1000 queries returned documents. The results in Table 4 are based on manual evaluation of all documents returned for the first 50 queries, and determination of whether they were multilingual dictionaries containing the indicated languages.

The baseline results are substantially higher than those for the synthetic dataset, almost certainly a direct result of the greater sophistication and optimisation of the Google search engine (including query log analysis, and link and anchor text analysis). Despite this, the results for our method are lower than those over the synthetic dataset, we suspect largely as a result of the style of queries we issue being so far removed from standard Google query patterns. Having said this, MAP scores of 0.32–0.92 suggest that the method is highly usable (i.e. at any given cutoff in the document ranking, an average of at least one in three documents is a genuine multilingual dictionary), and any non-dictionary documents returned by the method could easily be pruned by a lexicographer.

Lang	Dicts	MAP	Baseline
zh	16	0.55	0.19
es	17	0.92	0.13
ja	13	0.32	0.04
de	34	0.77	0.09
fr	36	0.77	0.08
it	23	0.69	0.11
ar	8	0.39	0.17
AVERAGE:	21.0	0.63	0.12

Table 4: Dictionary retrieval results over the open web for dictionaries containing English and each of the indicated languages (“Dicts” = the number of unique multilingual dictionaries retrieved for that language).

Among the 7 language pairs, en-es, en-de, en-fr and en-it achieved the highest MAP scores. In terms of unique lexical resources found with 50 queries, the most successful language pairs were en-fr, en-de and en-it.

6 Conclusions

We have described initial results for a method designed to automatically detect multilingual dictionaries on the web, and attained highly credible results over both a synthetic dataset and an experiment over the open web using a web search engine.

In future work, we hope to explore the ability of the method to detect domain-specific dictionaries (e.g. training over domain-specific dictionaries from other language pairs), and low-density languages where there are few dictionaries and Wikipedia articles to train the method on.

Acknowledgements

We wish to thank the anonymous reviewers for their valuable comments, and the Panlex developers for assistance with the dictionaries and experimental design. This research was supported by funding from the Group of Eight and the Australian Research Council.

References

- Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. 1993. Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, 22(2):207–216.
- Timothy Baldwin, Jonathan Pool, and Susan M. Colowick. 2010. PanLex and LEXTRACT: Translating all words of all languages of the world. In

- Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010), Demo Volume*, pages 37–40, Beijing, China.
- Jiang Chen and Jian-Yun Nie. 2000. Parallel web text mining for cross-language IR. In *Proceedings of Recherche d’Informations Assistée par Ordinateur 2000 (RIA0’2000)*, pages 62–77, Collège de France, France.
- ClueWeb09. 2009. The ClueWeb09 dataset. <http://lemurproject.org/clueweb09/>.
- Aidan Finn, Nicholas Kushmerick, and Barry Smyth. 2002. Genre classification and domain transfer for information filtering. In *Proceedings of the 24th European Conference on Information Retrieval (ECIR 2002)*, pages 353–362, Glasgow, UK.
- Freedict. 2011. Freedict dictionaries. <http://www.freedict.com>.
- Pascale Fung. 1998. A statistical view on bilingual lexicon extraction: From parallel corpora to non-parallel corpora. In *Proceedings of Association for Machine Translation in the Americas (AMTA 1998): Machine Translation and the Information Soup*, pages 1–17, Langhorne, USA.
- Indri. 2009. Indri search engine. <http://www.lemurproject.org/indri/>.
- David Kamholz and Jonathan Pool. to appear. PanLex: Building a resource for panlingual lexical translation. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC 2014)*, Reykjavik, Iceland.
- Batia Laufer and Linor Hadar. 1997. Assessing the effectiveness of monolingual, bilingual, and “bilingualised” dictionaries in the comprehension and production of new words. *The Modern Language Journal*, 81(2):189–196.
- Marco Lui, Jey Han Lau, and Timothy Baldwin. 2014. Automatic detection and language identification of multilingual documents. *Transactions of the Association for Computational Linguistics*, 2(Feb):27–40.
- Katsushi Matsuda and Toshikazu Fukushima. 1999. Task-oriented world wide web retrieval by document type classification. In *Proceedings of the 1999 ACM Conference on Information and Knowledge Management (CIKM 1999)*, pages 109–113, Kansas City, USA.
- MeCab. 2011. <http://mecab.googlecode.com>.
- I. Dan Melamed. 1996. Automatic construction of clean broad-coverage translation lexicons. In *Proceedings of the 2nd Conference of the Association for Machine Translation in the Americas (AMTA 1996)*, Montreal, Canada.
- Dragos Stefan Munteanu and Daniel Marcu. 2005. Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31(4):477–504.
- Masaaki Nagata, Teruka Saito, and Kenji Suzuki. 2001. Using the web as a bilingual dictionary. In *Proceedings of the ACL 2001 Workshop on Data-driven Methods in Machine Translation*, pages 1–8, Toulouse, France.
- Jian-Yun Nie. 2010. *Cross-language information retrieval*. Morgan and Claypool Publishers, San Rafael, USA.
- John M. Prager. 1999. Linguini: language identification for multilingual documents. In *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences (HICSS-32)*, Maui, USA.
- Philip Resnik and Noah A. Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380.
- Stephen Soderland, Christopher Lim, Mausam, Bo Qin, Oren Etzioni, and Jonathan Pool. 2009. Lemmatic machine translation. In *Proceedings of the Twelfth Machine Translation Summit (MT Summit XII)*, Ottawa, Canada.
- Nicholas Thieberger and Andrea L. Berez. 2012. Linguistic data management. In Nicholas Thieberger, editor, *The Oxford Handbook of Linguistic Fieldwork*. Oxford University Press, Oxford, UK.
- Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter for sighthan bake-off 2005. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, volume 171.
- Hiroshi Yamaguchi and Kumiko Tanaka-Ishii. 2012. Text segmentation by language using minimum description length. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 969–978, Jeju Island, Korea.
- Kun Yu and Junichi Tsujii. 2009. Bilingual dictionary extraction from Wikipedia. In *Proceedings of the Twelfth Machine Translation Summit (MT Summit XII)*, pages 379–386, Ottawa, Canada.
- Sven Meyer zu Eissen and Benno Stein. 2005. Genre classification of web pages. In *Proceedings of the 27th Annual German Conference in AI (KI 2005)*, pages 256–269, Ulm, Germany.

Automatic Detection of Cognates Using Orthographic Alignment

Alina Maria Ciobanu, Liviu P. Dinu

Faculty of Mathematics and Computer Science, University of Bucharest

Center for Computational Linguistics, University of Bucharest

`alina.ciobanu@my.fmi.unibuc.ro, ldinu@fmi.unibuc.ro`

Abstract

Words undergo various changes when entering new languages. Based on the assumption that these linguistic changes follow certain rules, we propose a method for automatically detecting pairs of cognates employing an orthographic alignment method which proved relevant for sequence alignment in computational biology. We use aligned subsequences as features for machine learning algorithms in order to infer rules for linguistic changes undergone by words when entering new languages and to discriminate between cognates and non-cognates. Given a list of known cognates, our approach does not require any other linguistic information. However, it can be customized to integrate historical information regarding language evolution.

1 Introduction

Cognates are words in different languages having the same etymology and a common ancestor. Investigating pairs of cognates is very useful in historical and comparative linguistics, in the study of language relatedness (Ng et al., 2010), phylogenetic inference (Atkinson et al., 2005) and in identifying how and to what extent languages change over time. In other several research areas, such as language acquisition, bilingual word recognition (Dijkstra et al., 2012), corpus linguistics (Simard et al., 1992), cross-lingual information retrieval (Buckley et al., 1997) and machine translation (Kondrak et al., 2003), the condition of common etymology is usually not essential and cognates are regarded as words with high cross-lingual meaning and orthographic or phonetic similarity.

The wide range of applications in which cognates prove useful attracted more and more at-

tention on methods for detecting such related pairs of words. This task is most challenging for resource-poor languages, for which etymologically related information is not accessible. Therefore, the research (Inkpen et al., 2005; Mulloni and Pekar, 2006; Hauer and Kondrak, 2011) focused on automatic identification of cognate pairs, starting from lists of known cognates.

In this paper, we propose a method for automatically determining pairs of cognates across languages. The proposed method requires a list of known cognates and, for languages for which additional linguistic information is available, it can be customized to integrate historical information regarding the evolution of the language. The rest of the paper is organized as follows: in Section 2 we present and analyze alternative methods and related work in this area. In Section 3 we introduce our approach for detection of cognates using orthographic alignment. In Section 4 we describe the experiments we conduct and we report and analyze the results, together with a comparison with previous methods. Finally, in Section 5 we draw the conclusions of our study and describe our plans for extending the method.

2 Related Work

There are three important aspects widely investigated in the task of cognate identification: semantic, phonetic and orthographic similarity. They were employed both individually (Simard et al., 1992; Inkpen et al., 2005; Church, 1993) and combined (Kondrak, 2004; Steiner et al., 2011) in order to detect pairs of cognates across languages. For determining semantic similarity, external lexical resources, such as WordNet (Fellbaum, 1998), or large corpora, might be necessary. For measuring phonetic and orthographic proximity of cognate candidates, string similarity metrics can be applied, using the phonetic or orthographic word forms as input. Various measures were investi-

gated and compared (Inkpen et al., 2005; Hall and Klein, 2010); Levenshtein distance (Levenshtein, 1965), XDice (Brew and McKelvie, 1996) and the longest common subsequence ratio (Melamed, 1995) are among the most frequently used metrics in this field. Gomes and Lopes (2011) proposed SpSim, a more complex method for computing the similarity of cognate pairs which tolerates learned transitions between words.

Algorithms for string alignment were successfully used for identifying cognates based on both their forms, orthographic and phonetic. Delmestri and Cristianini (2010) used basic sequence alignment algorithms (Needleman and Wunsch, 1970; Smith and Waterman, 1981; Gotoh, 1982) to obtain orthographic alignment scores for cognate candidates. Kondrak (2000) developed the ALINE system, which aligns words' phonetic transcriptions based on multiple phonetic features and computes similarity scores using dynamic programming. List (2012) proposed a framework for automatic detection of cognate pairs, LexStat, which combines different approaches to sequence comparison and alignment derived from those used in historical linguistics and evolutionary biology.

The changes undergone by words when entering from one language into another and the transformation rules they follow have been successfully employed in various approaches to cognate detection (Koehn and Knight, 2000; Mulloni and Pekar, 2006; Navlea and Todirascu, 2011). These orthographic changes have also been used in cognate production, which is closely related to the task of cognate detection, but has not yet been as intensively studied. While the purpose of cognate detection is to determine whether two given words form a cognate pair, the aim of cognate production is, given a word in a source language, to automatically produce its cognate pair in a target language. Beinborn et al. (2013) proposed a method for cognate production relying on statistical character-based machine translation, learning orthographic production patterns, and Mulloni (2007) introduced an algorithm for cognate production based on edit distance alignment and the identification of orthographic cues when words enter a new language.

3 Our Approach

Although there are multiple aspects that are relevant in the study of language relatedness, such

as orthographic, phonetic, syntactic and semantic differences, in this paper we focus only on lexical evidence. The orthographic approach relies on the idea that sound changes leave traces in the orthography and alphabetic character correspondences represent, to a fairly large extent, sound correspondences (Delmestri and Cristianini, 2010).

Words undergo various changes when entering new languages. We assume that rules for adapting foreign words to the orthographic system of the target languages might not have been very well defined in their period of early development, but they may have since become complex and probably language-specific. Detecting pairs of cognates based on etymology is useful and reliable, but, for resource-poor languages, methods which require less linguistic knowledge might be necessary. According to Gusfield (1997), an edit transcript (representing the conversion of one string to another) and an alignment are mathematically equivalent ways of describing relationships between strings. Therefore, because the edit distance was widely used in this research area and produced good results, we are encouraged to employ orthographic alignment for identifying pairs of cognates, not only to compute similarity scores, as was previously done, but to use aligned subsequences as features for machine learning algorithms. Our intuition is that inferring language-specific rules for aligning words will lead to better performance in the task of cognate identification.

3.1 Orthographic Alignment

String alignment is closely related to the task of sequence alignment in computational biology. Therefore, to align pairs of words we employ the Needleman-Wunsch global alignment algorithm (Needleman and Wunsch, 1970), which is mainly used for aligning sequences of proteins or nucleotides. Global sequence alignment aims at determining the best alignment over the entire length of the input sequences. The algorithm uses dynamic programming and, thus, guarantees to find the optimal alignment. Its main idea is that any partial path of the alignment along the optimal path should be the optimal path leading up to that point. Therefore, the optimal path can be determined by incremental extension of the optimal subpaths (Schuler, 2002). For orthographic alignment, we consider words as input sequences and we use a very simple substitution matrix, which

gives equal scores to all substitutions, disregarding diacritics (e.g., we ensure that e and \grave{e} are matched).

3.2 Feature Extraction

Using aligned pairs of words as input, we extract features around mismatches in the alignments. There are three types of mismatches, corresponding to the following operations: insertion, deletion and substitution. For example, for the Romanian word *exhaustiv* and its Italian cognate pair *esaustivo*, the alignment is as follows:

```

e x h a u s t i v -
e s - a u s t i v o

```

The first mismatch (between x and s) is caused by a substitution, the second mismatch (between h and $-$) is caused by a deletion from source language to target language, and the third mismatch (between $-$ and o) is caused by an insertion from source language to target language. The features we use are character n -grams around mismatches. We experiment with two types of features:

- i) n -grams around gaps, i.e., we account only for insertions and deletions;
- ii) n -grams around any type of mismatch, i.e., we account for all three types of mismatches.

The second alternative leads to better performance, so we account for all mismatches. As for the length of the grams, we experiment with $n \in \{1, 2, 3\}$. We achieve slight improvements by combining n -grams, i.e., for a given n , we use all i -grams, where $i \in \{1, \dots, n\}$. In order to provide information regarding the position of the features, we mark the beginning and the end of the word with a $\$$ symbol. Thus, for the above-mentioned pair of cognates, (*exhaustiv*, *esaustivo*), we extract the following features when $n = 2$:

```

x>s  ex>es  xh>s-
h>-  xh>s-  ha>-a
->o  v->vo  ->o$

```

For identical features we account only once. Therefore, because there is one feature ($xh>s-$) which occurs twice in our example, we have 8 features for the pair (*exhaustiv*, *esaustivo*).

3.3 Learning Algorithms

We use Naive Bayes as a baseline and we experiment with Support Vector Machines (SVMs) to

learn orthographic changes and to discriminate between pairs of cognates and non-cognates. We put our system together using the Weka workbench (Hall et al., 2009), a suite of machine learning algorithms and tools. For SVM, we use the wrapper provided by Weka for LibSVM (Chang and Lin, 2011). We use the radial basis function kernel (RBF), which can handle the case when the relation between class labels and attributes is non-linear, as it maps samples non-linearly into a higher dimensional space. Given two instances x_i and x_j , where $x_i \in \mathbb{R}^n$, the RBF kernel function for x_i and x_j is defined as follows:

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2), \gamma > 0,$$

where γ is a kernel parameter.

We split the data in two subsets, for training and testing, with a 3:1 ratio, and we perform grid search and 3-fold cross validation over the training set in order to optimize hyperparameters c and γ . We search over $\{1, 2, \dots, 10\}$ for c and over $\{10^{-5}, 10^{-4}, \dots, 10^4, 10^5\}$ for γ . The values which optimize accuracy on the training set are reported, for each pair of languages, in Table 3.

4 Experiments

4.1 Data

We apply our method on an automatically extracted dataset of cognates for four pairs of languages: Romanian-French, Romanian-Italian, Romanian-Spanish and Romanian-Portuguese. In order to build the dataset, we apply the methodology proposed by Ciobanu and Dinu (2014) on the DexOnline¹ machine-readable dictionary for Romanian. We discard pairs of words for which the forms across languages are identical (i.e., the Romanian word *matrice* and its Italian cognate pair *matrice*, having the same form), because these pairs do not provide any orthographic changes to be learned. For each pair of languages we determine a number of non-cognate pairs equal to the number of cognate pairs. Finally, we obtain 445 pairs of cognates for Romanian-French², 3,477 for Romanian-Italian, 5,113 for Romanian-Spanish and 7,858 for Romanian-Portuguese. Because we need sets of approximately equal size for

¹<http://dexonline.ro>

²The number of pairs of cognates is much lower for French than for the other languages because there are numerous Romanian words which have French etymology and, in this paper, we do not consider these words to be cognate candidates.

	1 st	2 nd	3 rd	4 th	5 th
IT	iu>io	un>on	l->le	t\$>-\$	-\$>e\$
FR	un>on	ne>n-	iu>io	ti>ti	e\$>-\$
ES	-\$>o\$	ti>ci	->ón	ie>ió	at>ad
PT	ie>ão	aç>aç	ti>çã	i\$>-\$	ã\$>a\$

Table 1: The most relevant orthographic cues for each pair of languages determined on the entire datasets using the χ^2 attribute evaluation method implemented in Weka.

	1 st	2 nd	3 rd	4 th	5 th
IT	-\$>e\$	-\$>o\$	ã\$>a\$	->re	ti>zi
FR	e\$>-\$	un>on	ne>n-	iu>io	ti>ti
ES	-\$>o\$	e\$>-\$	ti>ci	ã\$>a\$	at>ad
PT	-\$>o\$	ã\$>a\$	e\$>-\$	-\$>r\$	-\$>a\$

Table 2: The most frequent orthographic cues for each pair of languages determined on the cognate lists using the raw frequencies.

comparison across languages, we keep 400 pairs of cognates and 400 pairs of non-cognates for each pair of languages. In Tables 1 and 2 we provide, for each pair of languages, the five most relevant 2-gram orthographic changes, determined using the χ^2 distribution implemented in Weka, and the five most frequent 2-gram orthographic changes in the cognate pairs from our dataset³. None of the top ranked orthographic cues occurs at the beginning of the word, while many of them occur at the end of the word. The most frequent operation in Tables 1 and 2 is substitution.

4.2 Results Analysis

We propose a method for automatic detection of cognate pairs using orthographic alignment. We experiment with two machine-learning approaches: Naive Bayes and SVM. In Table 3 we report the results of our research. We report the n -gram values for which the best results are obtained and the hyperparameters for SVM, c and γ . The best results are obtained for French and Spanish, while the lowest accuracy is obtained for Portuguese. The SVM produces better results for all languages except Portuguese, where the accuracy is equal. For Portuguese, both Naive Bayes and SVM misclassify more non-cognates as cognates

³For brevity, we use in the tables the ISO 639-1 codes for language abbreviation. We denote pairs of languages by the target language, given the fact that Romanian is always the source language in our experiments.

than viceversa. A possible explanation might be the occurrence, in the dataset, of more remotely related words, which are not labeled as cognates. We plan to investigate this assumption and to apply the proposed method on other datasets in our future work.

4.3 Comparison with Previous Methods

We investigate the performance of the method we propose in comparison to previous approaches for automatic detection of cognate pairs based on orthographic similarity. We employ several orthographic metrics widely used in this research area: the edit distance (Levenshtein, 1965), the longest common subsequence ratio (Melamed, 1995) and the XDice metric (Brew and McKelvie, 1996)⁴. In addition, we use SpSim (Gomes and Lopes, 2011), which outperformed the longest common subsequence ratio and a similarity measure based on the edit distance in previous experiments. To evaluate these metrics on our dataset, we use the same train/test sets as we did in our previous experiments and we follow the strategy described in (Inkpen et al., 2005). First, we compute the pairwise distances between pairs of words for each orthographic metric individually, as a single feature⁵. In order to detect the best threshold for discriminating between cognates and non-cognates, we run a decision stump classifier (provided by Weka) on the training set for each pair of languages and for each metric. A decision stump is a decision tree classifier with only one internal node and two leaves corresponding to our two class labels. Using the best threshold value selected for each metric and pair of languages, we further classify the pairs of words in our test sets as cognates or non-cognates. In Table 4 we report the results for each approach. Our method performs better than the orthographic metrics considered as individual features. Out of the four similarity metrics, SpSim obtains, overall, the best performance. These results support the relevance of accounting for orthographic cues in cognate identification.

⁴We use normalized similarity metrics. For the edit distance, we subtract the normalized value from 1 in order to obtain similarity.

⁵SpSim cannot be computed directly, as the other metrics, so we introduce an additional step in which we use 1/3 of the training set (only cognates are needed) to learn orthographic changes. In order to maintain a stratified dataset, we discard an equal number of non-cognates in the training set and then we compute the distances for the rest of the training set and for the test set. We use the remaining of the initial training set for the next step of the procedure.

	Naive Bayes				SVM					
	P	R	A	n	P	R	A	n	c	γ
IT	0.72	0.93	79.0	1	0.76	0.92	81.5	1	1	0.10
FR	0.81	0.91	82.0	2	0.84	0.89	87.0	2	10	0.01
ES	0.79	0.92	84.0	1	0.85	0.88	86.5	2	4	0.01
PT	0.67	0.88	73.0	2	0.70	0.78	73.0	2	10	0.01

Table 3: Results for automatic detection of cognates using orthographic alignment. We report the precision (P), recall (R) and accuracy (A) obtained on the test sets and the optimal n -gram values. For SVM we also report the optimal hyperparameters c and γ obtained during cross-validation on the training sets.

	EDIT				LCSR				XDICE				SPSIM			
	P	R	A	t	P	R	A	t	P	R	A	t	P	R	A	t
IT	0.67	0.97	75.0	0.43	0.68	0.91	75.0	0.51	0.66	0.98	74.0	0.21	0.66	0.98	74.5	0.44
FR	0.76	0.93	82.0	0.30	0.76	0.90	81.5	0.42	0.77	0.79	78.0	0.26	0.86	0.83	85.0	0.59
ES	0.77	0.91	82.0	0.56	0.72	0.97	80.0	0.47	0.72	0.99	80.5	0.19	0.81	0.90	85.0	0.64
PT	0.62	0.99	69.5	0.34	0.59	0.99	65.5	0.34	0.57	0.99	63.5	0.10	0.62	0.97	69.0	0.39

Table 4: Comparison with previous methods for automatic detection of cognate pairs based on orthography. We report the precision (P), recall (R) and accuracy (A) obtained on the test sets and the optimal threshold t for discriminating between cognates and non-cognates.

5 Conclusions and Future Work

In this paper we proposed a method for automatic detection of cognates based on orthographic alignment. We employed the Needleman-Wunsch algorithm (Needleman and Wunsch, 1970) for sequence alignment widely-used in computational biology and we used aligned pairs of words to extract rules for lexical changes occurring when words enter new languages. We applied our method on an automatically extracted dataset of cognates for four pairs of languages.

As future work, we plan to extend our method on a few levels. In this paper we used a very simple substitution matrix for the alignment algorithm, but the method can be adapted to integrate historical information regarding language evolution. The substitution matrix for the alignment algorithm can be customized with language-specific information, in order to reflect the probability of a character to change into another. An important achievement in this direction belongs to Delmestri and Cristianini (2010), who introduced PAM-like matrices, linguistic-inspired substitution matrices which are based on information regarding orthographic changes. We plan to investigate the contribution of using this type of substitution matrices for our method.

We intend to investigate other approaches to string alignment, such as local alignment (Smith

and Waterman, 1981), and other learning algorithms for discriminating between cognates and non-cognates. We plan to extend our analysis with more language-specific features, where linguistic knowledge is available. First, we intend to use the part of speech as an additional feature. We assume that some orthographic changes are dependent on the part of speech of the words. Secondly, we want to investigate whether accounting for the common ancestor language influences the results. We are interested to find out if the orthographic rules depend on the source language, or if they are rather specific to the target language. Finally, we plan to make a performance comparison on cognate pairs versus word-etymon pairs and to investigate false friends (Nakov et al., 2007).

We further intend to adapt our method for cognate detection to a closely related task, namely cognate production, i.e., given an input word w , a related language L and a set of learned rules for orthographic changes, to produce the cognate pair of w in L .

Acknowledgements

We thank the anonymous reviewers for their helpful and constructive comments. The contribution of the authors to this paper is equal. Research supported by CNCS UEFISCDI, project number PN-II-ID-PCE-2011-3-0959.

References

- Quentin D. Atkinson, Russell D. Gray, Geoff K. Nicholls, and David J. Welch. 2005. From Words to Dates: Water into Wine, Mathemagic or Phylogenetic Inference? *Transactions of the Philological Society*, 103:193–219.
- Lisa Beinborn, Torsten Zesch, and Iryna Gurevych. 2013. Cognate Production using Character-based Machine Translation. In *Proceedings of the 6th International Joint Conference on Natural Language Processing, IJCNLP 2013*, pages 883–891.
- Chris Brew and David McKelvie. 1996. Word-Pair Extraction for Lexicography. In *Proceeding of Text, Speech and Dialogue, TSD 1996*, pages 45–55.
- Chris Buckley, Mandar Mitra, Janet A. Walz, and Claire Cardie. 1997. Using Clustering and Super-Concepts Within SMART: TREC 6. In *Proceedings of the 6th Text Retrieval Conference, TREC 1997*, pages 107–124.
- Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- Kenneth W. Church. 1993. Char align: A program for aligning parallel texts at the character level. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics, ACL 1993*, pages 1–8.
- Alina Maria Ciobanu and Liviu P. Dinu. 2014. Building a Dataset of Multilingual Cognates for the Romanian Lexicon. In *Proceedings of the 9th International Conference on Language Resources and Evaluation, LREC 2014*.
- Antonella Delmestri and Nello Cristianini. 2010. String Similarity Measures and PAM-like Matrices for Cognate Identification. *Bucharest Working Papers in Linguistics*, 12(2):71–82.
- Ton Dijkstra, Franc Grootjen, and Job Schepens. 2012. Distributions of Cognates in Europe as Based on Levenshtein Distance. *Bilingualism: Language and Cognition*, 15:157–166.
- Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, Massachusetts.
- Luís Gomes and José Gabriel Pereira Lopes. 2011. Measuring spelling similarity for cognate identification. In *Proceedings of the 15th Portuguese Conference on Progress in Artificial Intelligence, EPIA 2011*, pages 624–633. Software available at <http://research.variancia.com/spsim>.
- Osamu Gotoh. 1982. An improved algorithm for matching biological sequences. *Journal of Molecular Biology*, 162(3):705–708.
- Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences: computer science and computational biology*. Cambridge University Press New York, NY, USA.
- David Hall and Dan Klein. 2010. Finding Cognate Groups Using Phylogenies. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL 2010*, pages 1030–1039.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18. Software available at <http://www.cs.waikato.ac.nz/ml/weka>.
- Bradley Hauer and Grzegorz Kondrak. 2011. Clustering semantically equivalent words into cognate sets in multilingual lists. In *5th International Joint Conference on Natural Language Processing, IJCNLP 2011*, pages 865–873.
- Diana Inkpen, Oana Frunza, and Grzegorz Kondrak. 2005. Automatic Identification of Cognates and False Friends in French and English. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP 2005*, pages 251–257.
- Philipp Koehn and Kevin Knight. 2000. Estimating Word Translation Probabilities from Unrelated Monolingual Corpora Using the EM Algorithm. In *Proceedings of the 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence*, pages 711–715.
- Grzegorz Kondrak, Daniel Marcu, and Keven Knight. 2003. Cognates Can Improve Statistical Translation Models. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, HLT-NAACL 2003*, pages 46–48.
- Grzegorz Kondrak. 2000. A New Algorithm for the Alignment of Phonetic Sequences. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference, NAACL 2000*, pages 288–295.
- Grzegorz Kondrak. 2004. Combining Evidence in Cognate Identification. In *Proceedings of the 17th Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence*, pages 44–59.
- Vladimir I. Levenshtein. 1965. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics Doklady*, 10:707–710.
- Johann-Mattis List. 2012. LexStat: Automatic Detection of Cognates in Multilingual Wordlists. In *Proceedings of the EACL 2012 Joint Workshop of LINGVIS and UNCLH*, pages 117–125.

- Dan Melamed. 1995. Automatic Evaluation and Uniform Filter Cascades for Inducing N-Best Translation Lexicons. In *Proceedings of the 3rd Workshop on Very Large Corpora*.
- Andrea Mulloni and Viktor Pekar. 2006. Automatic detection of orthographic cues for cognate recognition. In *Proceedings of the 5th International Conference on Language Resources and Evaluation, LREC 2006*, pages 2387–2390.
- Andrea Mulloni. 2007. Automatic Prediction of Cognate Orthography Using Support Vector Machines. In *Proceedings of the 45th Annual Meeting of the ACL: Student Research Workshop, ACL 2007*, pages 25–30.
- Svetlin Nakov, Preslav Nakov, and Elena Paskaleva. 2007. Cognate or False Friend? Ask the Web! In *Proceedings of the RANLP 2007 Workshop "Acquisition and Management of Multilingual Lexicons"*, pages 55–62.
- Mirabela Navlea and Amalia Todirascu. 2011. Using Cognates in a French-Romanian Lexical Alignment System: A Comparative Study. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP 2011*, pages 247–253.
- Saul B. Needleman and Christian D. Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443 – 453.
- Ee-Lee Ng, Beatrice Chin, Alvin W. Yeo, and Bali Ranaivo-Malançon. 2010. Identification of Closely-Related Indigenous Languages: An Orthographic Approach. *Int. J. of Asian Lang. Proc.*, 20(2):43–62.
- Gregory D. Schuler. 2002. Sequence Alignment and Database Searching. *Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins*, 43. A. D. Baxevanis and B. F. F. Ouellette, John Wiley & Sons, Inc., New York, USA.
- Michel Simard, George F. Foster, and Pierre Isabelle. 1992. Using Cognates to Align Sentences in Bilingual Corpora. In *Proceedings of the 4th International Conference on Theoretical and Methodological Issues in Machine Translation*.
- Temple F. Smith and Michael S. Waterman. 1981. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197.
- Lydia Steiner, Peter F. Stadler, and Michael Cysouw. 2011. A pipeline for computational historical linguistics. *Language Dynamics and Change*, 1(1):89–127.

Automatically constructing Wordnet synsets

Khang Nhut Lam, Feras Al Tarouti and Jugal Kalita

Computer Science department

University of Colorado

1420 Austin Bluffs Pkwy, Colorado Springs, CO 80918, USA

{klam2, faltarou, jkalita}@uccs.edu

Abstract

Manually constructing a Wordnet is a difficult task, needing years of experts' time. As a first step to automatically construct full Wordnets, we propose approaches to generate Wordnet synsets for languages both resource-rich and resource-poor, using publicly available Wordnets, a machine translator and/or a single bilingual dictionary. Our algorithms translate synsets of existing Wordnets to a target language T , then apply a ranking method on the translation candidates to find best translations in T . Our approaches are applicable to any language which has at least one existing bilingual dictionary translating from English to it.

1 Introduction

Wordnets are intricate and substantive repositories of lexical knowledge and have become important resources for computational processing of natural languages and for information retrieval. Good quality Wordnets are available only for a few "resource-rich" languages such as English and Japanese. Published approaches to automatically build new Wordnets are manual or semi-automatic and can be used only for languages that already possess some lexical resources.

The Princeton Wordnet (PWN) (Fellbaum, 1998) was painstakingly constructed manually over many decades. Wordnets, except the PWN, have been usually constructed by one of two approaches. The first approach translates the PWN to T (Bilgin et al., 2004), (Barbu and Mititelu, 2005), (Kaji and Watanabe, 2006), (Sagot and Fišer, 2008), (Saveski and Trajkovsk, 2010) and (Oliver and Climent, 2012); while the second approach builds a Wordnet in T , and then aligns it with the PWN by generating translations (Gu-

nawan and Saputra, 2010). In terms of popularity, the first approach dominates over the second approach. Wordnets generated using the second approach have different structures from the PWN; however, the complex agglutinative morphology, culture specific meanings and usages of words and phrases of target languages can be maintained. In contrast, Wordnets created using the first approach have the same structure as the PWN.

One of our goals is to automatically generate high quality synsets, each of which is a set of cognitive synonyms, for Wordnets having the same structure as the PWN in several languages. Therefore, we use the first approach to construct Wordnets. This paper discusses the first step of a project to automatically build core Wordnets for languages with low amounts of resources (viz., Arabic and Vietnamese), resource-poor languages (viz., Assamese) or endangered languages (viz., Dimasa and Karbi)¹. The sizes and the qualities of freely existing resources, if any, for these languages vary, but are not usually high. Hence, our second goal is to use a limited number of freely available resources in the target languages as input to our algorithms to ensure that our methods can be felicitously used with languages that lack much resource. In addition, our approaches need to have a capability to reduce noise coming from the existing resources that we use. For translation, we use a free machine translator (MT) and restrict ourselves to using it as the only "dictionary" we can have. For research purposes, we have obtained free access to the Microsoft Translator, which supports translations among 44 languages. In particular, given public Wordnets aligned to the PWN (such as the FinnWordNet (FWN) (Lindén, 2010) and the Japanese WordNet (JWN) (Isahara et al., 2008)) and the Microsoft Translator, we build Wordnet synsets for *arb*, *asm*, *dis*, *ajz* and *vie*.

¹ISO 693-3 codes of Arabic, Assamese, Dimasa, Karbi and Vietnamese are *arb*, *asm*, *dis*, *ajz* and *vie*, respectively.

2 Proposed approaches

In this section, we propose approaches to create Wordnet synsets for a target languages T using existing Wordnets and the MT and/or a single bilingual dictionary. We take advantage of the fact that every synset in PWN has a unique *offset-POS*, referring to the offset for a synset with a particular part-of-speech (POS) from the beginning of its data file. Each synset may have one or more words, each of which may be in one or more synsets. Words in a synset have the same sense. The basic idea is to extract corresponding synsets for each *offset-POS* from existing Wordnets linked to PWN, in several languages. Next, we translate extracted synsets in each language to T to produce so-called *synset candidates* using MT. Then, we apply a ranking method on these candidates to find the correct words for a specific *offset-POS* in T .

2.1 Generating synset candidates

We propose three approaches to generate synset candidates for each *offset-POS* in T .

2.1.1 The direct translation (DR) approach

The first approach directly translates synsets in PWN to T as in Figure 1.

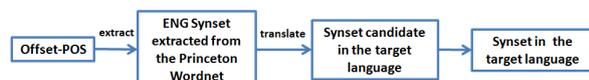


Figure 1: The DR approach to construct Wordnet synsets in a target language T .

For each *offset-POS*, we extract words in that synset from the PWN and translate them to the target language to generate translation candidates.

2.1.2 Approach using intermediate Wordnets (IW)

To handle ambiguities in synset translation, we propose the IW approach as in Figure 2. Publicly available Wordnets in various languages, which we call intermediate Wordnets, are used as resources to create synsets for Wordnets. For each *offset-POS*, we extract its corresponding synsets from intermediate Wordnets. Then, the extracted synsets, which are in different languages, are translated to T using MT to generate synset candidates. Depending on which Wordnets are used and the number of intermediate Wordnets, the number of candidates in each synset and the number of synsets in the new Wordnets change.

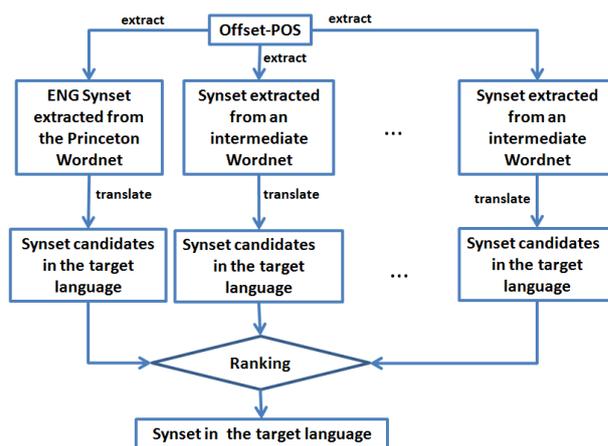


Figure 2: The IW approach to construct Wordnet synsets in a target language T

2.1.3 Approach using intermediate Wordnets and a dictionary (IWND)

The IW approach for creating Wordnet synsets decreases ambiguities in translations. However, we need more than one bilingual dictionary from each intermediate languages to T . Such dictionaries are not always available for many languages, especially the ones that are resource poor. The IWND approach is like the IW approach, but instead of translating immediately from the intermediate languages to the target language, we translate synsets extracted from intermediate Wordnets to English (*eng*), then translate them to the target language. The IWND approach is presented in Figure 3.

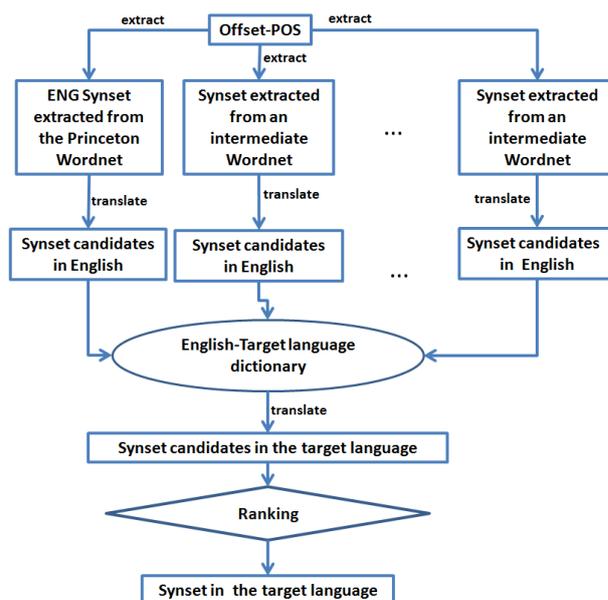


Figure 3: The IWND approach to construct Wordnet synsets

2.2 Ranking method

For each of *offset-POS*, we have many translation candidates. A translation candidate with a higher rank is more likely to become a word belonging to the corresponding *offset-POS* of the new Wordnet in the target language. Candidates having the same ranks are treated similarly. The rank value in the range 0.00 to 1.00. The rank of a word w , the so-called $rank_w$, is computed as below.

$$rank_w = \frac{occur_w}{numCandidates} * \frac{numDstWordnets}{numWordnets}$$

where:

- $numCandidates$ is the total number of translation candidates of an *offset-POS*
- $occur_w$ is the occurrence count of the word w in the $numCandidates$
- $numWordnets$ is the number of intermediate Wordnets used, and
- $numDstWordnets$ is the number of distinct intermediate Wordnets that have words translated to the word w in the target language.

Our motivation for this rank formula is the following. If a candidate has a higher occurrence count, it has a greater chance to become a correct translation. Therefore, the occurrence count of each candidate needs to be taken into account. We normalize the occurrence count of a word by dividing it by $numCandidates$. In addition, if a candidate is translated from different words having the same sense in different languages, this candidate is more likely to be a correct translation. Hence, we multiply the first fraction by $numDstWordnets$. To normalize, we divide results by the number of intermediate Wordnet used.

For instance, in our experiments we use 4 intermediate Wordnets, viz., PWN, FWN, JWN and WOLF Wordnet (WWN) (Sagot and Fišer, 2008). The words in the *offset-POS* "00006802-v" obtained from all 4 Wordnets, their translations to *arb*, the occurrence count and the rank of each translation are presented in the second, the fourth and the fifth columns, respectively, of Figure 4.

2.3 Selecting candidates based on ranks

We separate candidates based on three cases as below.

Case 1: A candidate w has the highest chance to become a correct word belonging to a specific synset in the target language if its rank is 1.0. This means that all intermediate Wordnets contain the synset having a specific *offset-POS* and all words belonging to these synsets are translated to the

Words	Cand.	TL	Occur	Rank
chuff ^A	شوف	shwf	1	0.036
huff ^A	هوف	hwf	1	0.036
puff ^A	نفخة	nfkhh	2	0.143
puuskutta ^B	بوسكوتا	bwvskwta	1	0.036
puhkua ^B	بوهكوا	bwhkwa	1	0.036
läähättää ^B	أنفاسها	anfasoha	1	0.036
bouffée ^C	نفخة	nfkhh	2	0.143

Figure 4: Example of calculating the ranks of candidates translated from words belonging to the *offset-POS* "00006802-v" in 4 Wordnets: PWN, FWN, JWN and WWN. The $word^A$, $word^B$ and $word^C$ are obtained from PWN, FWN and WWN, respectively. The JWN does not contain this *offset-POS*. *TL* presents transliterations of the words in *arb*. The $numWordnets$ is 4 and the $numCandidates$ is 7. The rank of each candidate is shown in the last column of Figure 4.

same word w . The more the number of intermediate Wordnets used, the higher the chance the candidate with the rank of 1.0 has to become the correct translation. Therefore, we accept all translations that satisfy this criterion. An example of this scenario is presented in Figure 5.

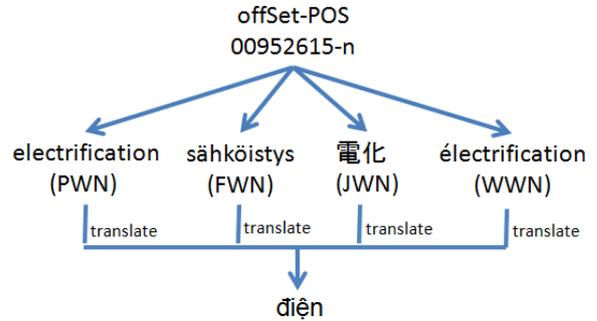


Figure 5: Example of Case 1: Using the IW approach with four intermediate Wordnets, PWN, FWN, JWN and WWN. All words belonging to the *offSet-POS* "00952615-n" in all 4 Wordnets are translated to the same word "điện" in *vie*. The word "điện" is accepted as the correct word belonging to the *offSet-POS* "00952615-n" in the Vietnamese Wordnet we create.

Case 2: If an *offSet-POS* does not have candidates having the rank of 1.0, we accept the candidates having the greatest rank. Figure 6 shows the example of the second scenario.

Case 3: If all candidates of an *offSet-POS* has the same rank which is also the greatest rank, we

Wordnet	Words	Cand.	Rank
PWN	send	gửi	0.67
PWN	send out	gửi	0.67
FWN	lähetää	gửi	0.67
WWN	transmettre	truyền tải	0.06
WWN	virer	chuyển giao	0.06
WWN	envoyer	gửi	0.67

Figure 6: Example of Case 2: Using the IW approach with three intermediate Wordnets, PWN, FWN and WWN. For the *offSet-POS* "01437254-v", there is no candidate with the rank of 1.0. The highest rank of the candidates in "vie" is 0.67 which is the word gửi. We accept "gửi" as the correct word in the *offSet-POS* "01437254-v" in the Vietnamese Wordnet we create.

skip these candidates. Table 1 gives an example of the last scenario.

Wordnet	Words	Cand.	Rank
PWN	act	hành động	0.33
PWN	behave	hoạt động	0.33
FWN	do	làm	0.33

Table 1: Example of Case 3: Using the DR approach. For the *offSet-POS* "00010435-v", there is no candidate with the rank of 1.0. The highest rank of the candidates in *vie* is 0.33. All of 3 candidates have the rank as same as the highest rank. Therefore, we do not accept any candidate as the correct word in the *offSet-POS* "00010435-v" in the Vietnamese Wordnet we create.

3 Experiments

3.1 Publicly available Wordnets

The PWN is the oldest and the biggest available Wordnet. It is also free. Wordnets in many languages are being constructed and developed². However, only a few of these Wordnets are of high quality and free for downloading. The EuroWordnet (Vossen, 1998) is a multilingual database with Wordnets in European languages (e.g., Dutch, Italian and Spanish). The AsianWordnet³ provides a platform for building and sharing Wordnets for Asian languages (e.g., Mongolian, Thai and Vietnamese). Unfortunately, the progress in building most of these Wordnets is slow and they are far from being finished.

²http://www.globalwordnet.org/gwa/Wordnet_table.html

³<http://www.asianwordnet.org/progress>

In our current experiments as mentioned earlier, we use the PWN and other Wordnets linked to the PWN 3.0 provided by the Open Multilingual Wordnet⁴ project (Bond and Foster, 2013): WWN, FWN and JWN. Table 2 provides some details of the Wordnets used.

Wordnet	Synsets	Core
JWN	57,179	95%
FWN	116,763	100%
PWN	117,659	100%
WWN	59,091	92%

Table 2: The number of synsets in the Wordnets linked to the PWN 3.0 are obtained from the Open Multilingual Wordnet, along with the percentage of synsets covered from the semi-automatically compiled list of 5,000 "core" word senses in PWN. Note that synsets which are not linked to the PWN are not taken into account.

For languages not supported by MT, we use three additional bilingual dictionaries: two dictionaries *Dict(eng,ajz)* and *Dict(eng,dis)* provided by Xobdo⁵; one *Dict(eng,asm)* created by integrating two dictionaries *Dict(eng,asm)* provided by Xobdo and Panlex⁶. The dictionaries are of varying qualities and sizes. The total number of entries in *Dict(eng,ajz)*, *Dict(eng,asm)* and *Dict(eng,dis)* are 4682, 76634 and 6628, respectively.

3.2 Experimental results and discussion

As previously mentioned, our primary goal is to build high quality synsets for Wordnets in languages with low amount of resources: *ajz*, *asm*, *arb*, *dis* and *vie*. The number of Wordnet synsets we create for *arb* and *vie* using the DR approach and the coverage percentage compared to the PWN synsets are 4813 (4.10%) and 2983 (2.54%), respectively. The number of synsets for each Wordnet we create using the IW approach with different numbers of intermediate Wordnets and the coverage percentage compared to the PWN synsets are presented in Table 3.

For the IWND approach, we use all 4 Wordnets as intermediate resources. The number of Wordnet synsets we create using the IWND approach are presented in Table 4. We only construct Wordnet synsets for *ajz*, *asm* and *dis* using the IWND ap-

⁴<http://compling.hss.ntu.edu.sg/omw/>

⁵<http://www.xobdo.org/>

⁶<http://panlex.org/>

App.	Lang.	WNs	Synsets	% coverage
IW	arb	2	48,245	41.00%
IW	vie	2	42,938	36.49%
IW	arb	3	61,354	52.15%
IW	vie	3	57,439	48.82%
IW	arb	4	75,234	63.94%
IW	vie	4	72,010	61.20%

Table 3: The number of Wordnet synsets we create using the IW approach. *WNs* is the number of intermediate Wordnets used: 2: PWN and FWN, 3: PWN, FWN and JWN and 4: PWN, FWN, JWN and WWN.

proach because these languages are not supported by MT.

App.	Lang.	Synsets	% coverage
IWND	ajz	21,882	18.60%
IWND	arb	70,536	59.95%
IWND	asm	43,479	36.95%
IWND	dis	24,131	20.51%
IWND	vie	42,592	36.20%

Table 4: The number of Wordnets synsets we create using the IWND approach.

Finally, we combine all of the Wordnet synsets we create using different approaches to generate the final Wordnet synsets. Table 5 presents the final number of Wordnet synsets we create and their coverage percentage.

Lang.	Synsets	% coverage
ajz	21,882	18.60%
arb	76,322	64.87%
asm	43,479	36.95%
dis	24,131	20.51%
vie	98,210	83.47%

Table 5: The number and the average score of Wordnets synsets we create.

Evaluations were performed by volunteers who use the language of the Wordnet as mother tongue. To achieve reliable judgment, we use the same set of 500 *offSet-POSs*, randomly chosen from the synsets we create. Each volunteer was requested to evaluate using a 5-point scale – 5: excellent, 4: good, 3: average, 2: fair and 1: bad. The average score of Wordnet synsets for *arb*, *asm* and *vie* are 3.82, 3.78 and 3.75, respectively. We notice that the Wordnet synsets generated using the IW approach with all 4 intermediate Wordnets have the highest average score: 4.16/5.00 for *arb* and

4.26/5.00 for *vie*. We are in the process of finding volunteers to evaluate the Wordnet synsets for *ajz* and *dis*.

It is difficult to compare Wordnets because the languages involved in different papers are different, the number and quality of input resources vary and the evaluation methods are not standard. However, for the sake of completeness, we make an attempt at comparing our results with published papers. Although our score is not in terms of percentage, we obtain the average score of 3.78/5.00 (or informally and possibly incorrectly, 75.60% precision) which we believe it is better than 55.30% obtained by (Bond et al., 2008) and 43.20% obtained by (Charoenporn et al., 2008). In addition, the average coverage percentage of all Wordnet synsets we create is 44.85% which is better than 12% in (Charoenporn et al., 2008) and 33276 synsets (\simeq 28.28%) in (Saveski and Trajkovsk, 2010).

The previous studies need more than one dictionary to translate between a target language and intermediate-helper languages. For example, to create the JWN, (Bond et al., 2008) needs the Japanese-Multilingual dictionary, Japanese-English lexicon and Japanese-English life science dictionary. For *asm*, there are a number of Dict(eng,asm); to the best of our knowledge only two online dictionaries, both between *eng* and *asm*, are available. The IWND approach requires only one input dictionary between a pair of languages. This is a strength of our method.

4 Conclusion and future work

We present approaches to create Wordnet synsets for languages using available Wordnets, a public MT and a single bilingual dictionary. We create Wordnet synsets with good accuracy and high coverage for languages with low resources (*arb* and *vie*), resource-poor (*asm*) and endangered (*ajz* and *dis*). We believe that our work has the potential to construct full Wordnets for languages which do not have many existing resources. We are in the process of creating a Website where all Wordnet synsets we create will be available, along with a user friendly interface to give feedback on individual entries. We will solicit feedback from communities that use these languages as mother-tongue. Our goal is to use this feedback to improve the quality of the Wordnet synsets. Some of Wordnet synsets we created can be downloaded from <http://cs.uccs.edu/~linclab/projects.html>.

References

- Antoni Oliver and Salvador Climent. 2012. Parallel corpora for Wordnet construction: Machine translation vs. automatic sense tagging. In *Proceedings of the 13th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing)*, volume part II, pages 110-121, New Delhi, India, March.
- Benoît Sagot and Darja Fišer. 2008. Building a free French Wordnet from multilingual resources. In *Proceedings of the Ontolex 2008 Workshop*, Marrakech, Morocco, May.
- Fellbaum, Christiane. 1998. *Wordnet: An electronic lexical database*. MIT Press, Cambridge, Massachusetts, USA.
- Francis Bond and Ryan Foster. 2013. Linking and extending an open multilingual Wordnet. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1352–1362, Sofia, Bulgaria, August.
- Francis Bond, Hitoshi Isahara, Kyoko Kanzaki and Kiyotaka Uchimoto. 2008. Boot-strapping a Wordnet using multiple existing Wordnets. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC)*, pages 1619–1624, Genoa, Italy, May.
- Eduard Barbu and Verginica Barbu Mititelu. 2005. Automatic building of Wordnets. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, Borovets, Bulgaria, September.
- Gunawan and Andy Saputra. 2010. Building synsets for Indonesian Wordnet with monolingual lexical resources. In *Proceedings of the International Conference on Asian Language Processing (IALP)*, pages 297–300, Harbin, China, December.
- Hiroyuki Kaji and Mariko Watanabe. 2006. Automatic construction of Japanese Wordnet. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*, pages 1262–1267, Genoa, Italy, May.
- Hitoshi Isahara, Francis Bond, Kiyotaka Uchimoto, Masao Utiyama and Kyoko Kanzaki. 2008. Development of Japanese Wordnet. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC)*, pages 2420–2423, Marrakech, Morocco, May.
- Krister Lindén and Laur Carlson. 2010. FinnWordnet - WordNet påfinska via översättning, *LexicoNordica. Nordic Journal of Lexicography*, 17:119–140.
- Martin Saveski and Igor Trajkovsk. 2010. Automatic construction of Wordnets by using machine translation and language modeling. In *Proceedings of the 13th Multiconference Information Society*, Ljubljana, Slovenia.
- Orhan Bilgin, Özlem Çentinoğlu and Kemal Oflazer. 2004. Building a Wordnet for Turkish. *Romanian Journal of Information Science and Technology*, 7(1-2): 163–172.
- Piek Vossen. 1998. *A multilingual database with lexical semantic networks*. Kluwer Academic Publishers, Dordrecht, Netherlands.
- Thatsanee Charoenporn, Virach Sornlertlamvanich, Chumpol Mokarat and Hitoshi Isahara. 2008. Semi-automatic compilation of Asian Wordnet, In *Proceedings of the 14th Annual Meeting of the Association for Natural Language Processing*, pages 1041–1044, Tokyo, Japan.

Constructing a Turkish-English Parallel TreeBank

Olcay Taner Yıldız[†], Ercan Solak[†], Onur Görgün^{†,††} and Razieh Ehsani[†]

[†] Işık University, Istanbul, Turkey

^{††} Alcatel Lucent Teletaş Telekomünikasyon A.Ş., Istanbul, Turkey

{olcaytaner, ercan, razieh.ehsani}@isikun.edu.tr

onur.gorgun@alcatel-lucent.com

Abstract

In this paper, we report our preliminary efforts in building an English-Turkish parallel treebank corpus for statistical machine translation. In the corpus, we manually generated parallel trees for about 5,000 sentences from Penn Treebank. English sentences in our set have a maximum of 15 tokens, including punctuation. We constrained the translated trees to the reordering of the children and the replacement of the leaf nodes with appropriate glosses. We also report the tools that we built and used in our tree translation task.

1 Introduction

Turkish is an agglutinative and morphologically rich language with a free constituent order. Although statistical NLP research on Turkish has taken significant steps in recent years, much remains to be done. Especially for the annotated corpora, Turkish is still behind similar languages such as Czech, Finnish, or Hungarian. For example, EuroParl corpus (Koehn, 2002), one of the biggest parallel corpora in statistical machine translation, contains 22 languages (but not Turkish). Although there exist some recent works to produce parallel corpora for Turkish-English pair, the produced corpus is only applicable for phrase-based training (Yeniterzi and Oflazer, 2010; El-Kahlout, 2009).

In recent years, many efforts have been made to annotate parallel corpora with syntactic structure to build parallel treebanks. A parallel treebank is a parallel corpus where the sentences in each language are syntactically (if necessary morphologically) annotated, and the sentences and words are aligned. In the parallel treebanks, the syntactic annotation usually follows constituent and/or dependency structure. Well-known parallel treebank efforts are

- Prague Czech-English dependency treebank annotated with dependency structure (Cmejrek et al., 2004)
- English-German parallel treebank, annotated with POS, constituent structures, functional relations, and predicate-argument structures (Cyrus et al., 2003)
- Linköping English-Swedish parallel treebank that contains 1,200 sentences annotated with POS and dependency structures (Ahrenberg, 2007)
- Stockholm multilingual treebank that contains 1,000 sentences in English, German and Swedish annotated with constituent structure (Gustafson-Capkova et al., 2007)

In this study, we report our preliminary efforts in constructing an English-Turkish parallel treebank corpus for statistical machine translation. Our approach converts English parse trees into equivalent Turkish parse trees by applying several transformation heuristics. The main components of our strategy are (i) tree permutation, where we permute the children of a node; and (ii) leaf replacement, where we replace English word token at a leaf node.

This paper is organized as follows: In Section 2, we give the literature review for parallel treebank construction efforts in Turkish. In Section 3, we give a very brief overview on Turkish syntax. We give the details of our corpus construction strategy in Section 4 and explain our transformation heuristics in Section 5. Finally, we conclude in Section 6.

2 Literature Review

Turkish Treebank creation efforts started with the METU-Sabancı dependency Treebank. METU-Sabancı Treebank explicitly represents the head-dependent relations and functional categories. In

order to adapt the corpus written in 1990's Turkish to further studies, a subset of 7.262 sentences of the corpus was manually annotated morphologically and syntactically (Atalay et al., 2003). METU-Sabancı Treebank is then used in many Turkish NLP studies (Eryigit and Oflazer, 2006; Yuret, 2006; Riedel et al., 2006; Ruket and Baldrige, 2006; Eryigit et al., 2006; Eryigit et al., 2008).

METU-Sabancı Treebank is also subject to transformation efforts from dependency-structure to constituency-structure. Combinatory Categorical Grammar (CCG) is extracted from the METU-Sabancı Treebank with annotation of lexical categories (Cakici, 2005). Sub-lexical units revealing the internal structure of the words are used to generate a Lexical Grammar Formalism (LGF) for Turkish with the help of finite state machines (Cetinoglu and Oflazer, 2006; Cetinoglu and Oflazer, 2009).

Swedish-Turkish parallel treebank is the first parallel Treebank effort for Turkish (Megyesi et al., 2008). The treebank is a balanced syntactically annotated corpus containing both fiction and technical documents. In total, it consists of approximately 160,000 tokens in Swedish and 145,000 in Turkish. Parallel texts are linguistically annotated using different layers from part of speech tags and morphological features to dependency annotation.

English-Swedish-Turkish parallel treebank (Megyesi et al., 2010), mainly the successor of the Swedish-Turkish parallel treebank, consists of approximately 300,000 tokens in Swedish, 160,000 in Turkish and 150,000 in English. The majority of the original text is written in Swedish and translated to Turkish and/or English. For the syntactic description, dependency structure is chosen instead of the constituent structure. All data is automatically annotated with syntactic tags using MaltParser (Nivre et al., 2006a). MaltParser is trained on the Penn Treebank for English, on the Swedish treebank Talbanken05 (Nivre et al., 2006b), and on the METU-Sabancı Turkish Treebank (Atalay et al., 2003), respectively.

ParGram parallel treebank (Sulger et al., 2013) is a joint effort for the construction of a parallel treebank involving ten languages (English, Georgian, German, Hungarian, Indonesian, Norwegian, Polish, Turkish, Urdu, Wolof) from six language families. The treebank is based on deep Lexical-Functional Grammars that were devel-

oped within the framework of the Parallel Grammar effort. ParGram treebank allows for the alignment of sentences at several levels: dependency structures, constituency structures and POS information.

3 Turkish syntax

Turkish is an agglutinative language with rich derivational and inflectional morphology through suffixes. Word forms usually have a complex yet fairly regular morphotactics.

Turkish sentences have an unmarked SOV order. However, depending on the discourse, constituents can be scrambled to emphasize, topicalize and focus certain elements. Case markings identify the syntactic functions of the constituents, (Kornfilt, 1997).

4 Corpus construction strategy

In order to constrain the syntactic complexity of the sentences in the corpus, we selected from the Penn Treebank II 9560 trees which contain a maximum of 15 tokens. These include 8660 trees from the training set of the Penn Treebank, 360 trees from its development set and 540 trees from its test set. In the first phase of our work, we translated 4247 trees of the training set and all of those in the development and the test sets.

4.1 Tools

Manual annotation is an error prone task. From simple typos to disagreements among annotators, the range of errors is fairly large. An annotation tool needs to help reduce these errors and help the annotator locate them when they occur. Moreover, the tool needs to present the annotator with a visual tree that is both easy to understand and manipulate for the translation task.

We built a range of custom tools to display, manipulate and save annotated trees in the treebank. The underlying data structure is still textual and uses the standard Treebank II style of syntactic bracketing.

We also implemented a simple statistical helper function within the tool. When translating an English word to a gloss in Turkish, the translator may choose from a list of glosses sorted according their likelihood calculated over their previous uses in similar cases. Thus, as the corpus grows in size, the translators use the leverage of their previous choices.

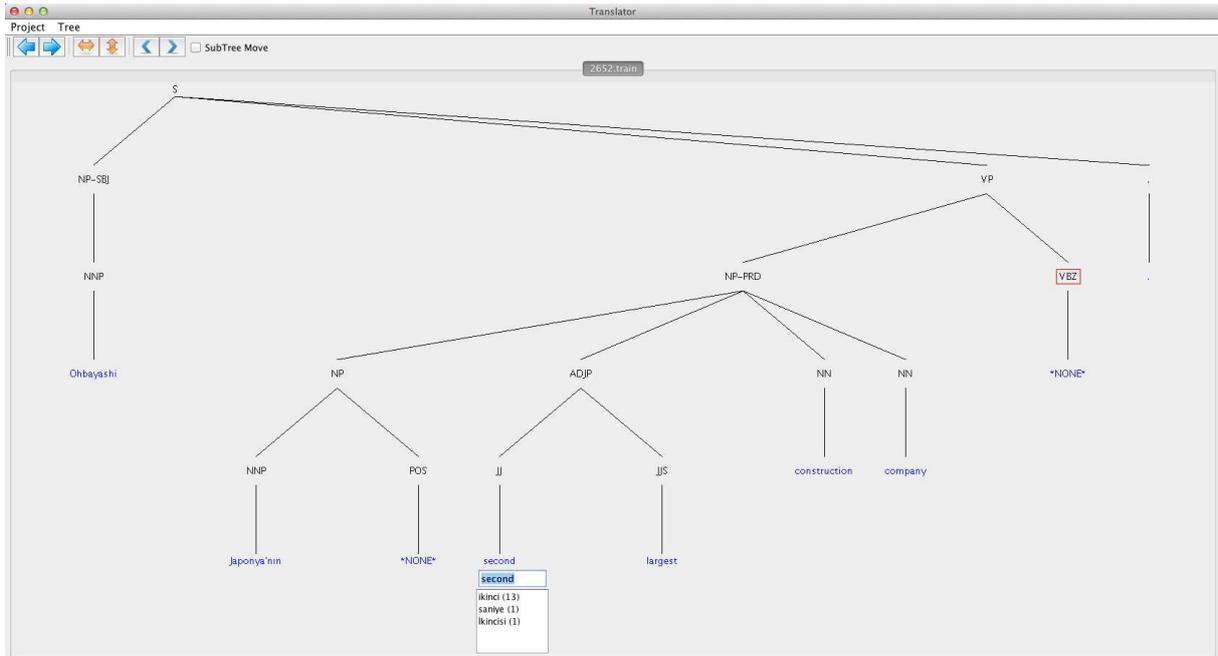


Figure 1: A screenshot of the tree translation tool

Figure 1 shows a screenshot of our tree translation tool.

4.2 Tree permutation

In translating an English syntactic tree, we confine ourselves to two operations. We can permute the children of a node and we can replace the English word token at a leaf node. No other modification of the tree is allowed. In particular, we use the same set of tags and predicate labels in the non-leaf nodes and do not use new tags for the Turkish trees. Adding or deleting nodes are not allowed either.

This might seem like a rather restrictive view of translation. Indeed, it is very easy to construct pairs of translated sentences which involve operations outside our restricted set when transformed into each other.

However, we use the following method to alleviate the restrictions of the small set of operations.

We use the **NONE** tag when we can not use any direct gloss for an English token. In itself, this operation corresponds to effectively mapping an English token to a null token. However, when we use the **NONE** tag, permute the nodes and choose the full inflected forms of the glosses in the Turkish tree, we have a powerful method to convert subtrees to an inflected word. The tree in Figure 2. illustrates this. Note that the POS tag sequence VP-RB-MD-PRP in the Turkish sentence

corresponds to the morphological analysis “geç-NEG-FUT-2SG” of the verb “geçmeyeceksin”. In general, we try to permute the nodes so as to correspond to the order of inflectional morphemes in the chosen gloss.

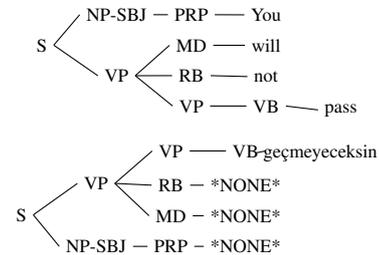


Figure 2: The permutation of the nodes and the replacement of the leaves by the glosses or **NONE**.

5 Transformation heuristics

When we have a sufficiently rich corpus of parallel trees, our next step is to train a SMT learner to imitate the human translator who operates under our restricted set of operations. Naturally, human translators often base their transformation decisions on the whole tree. Still, having a common set of rules and heuristics helps the translators in both consistency and speed. In the following, we illustrate these heuristics.

5.1 Constituent and morpheme order

Majority of unmarked Turkish sentences have the SOV order. When translating English trees, we permute its shallow subtrees to reflect the change of constituent order in Turkish.

Also, the agglutinative suffixes of Turkish words dictate the order when permuting the constituents which correspond to prepositions and particles.

The semantic aspects expressed by prepositions, modals, particles and verb tenses in English in general correspond to specific morphemes attached to the corresponding word stem. For example, “Ali/NNP will/MD sit/VB on/IN a/DT chair/NN” is literally translated as

Ali bir sandalye-ye otur-acak.

Ali a chair-DAT sit-FUT.

If we embed a constituent in the morphemes of a Turkish stem, we replace the English constituent leaf with *NONE*.

In some cases, the personal pronouns acting as subjects are naturally embedded in the verb inflection. In those cases, pronoun in the original tree is replaced with *NONE* and its subtree is moved to after the verb phrase. See Figure 3.

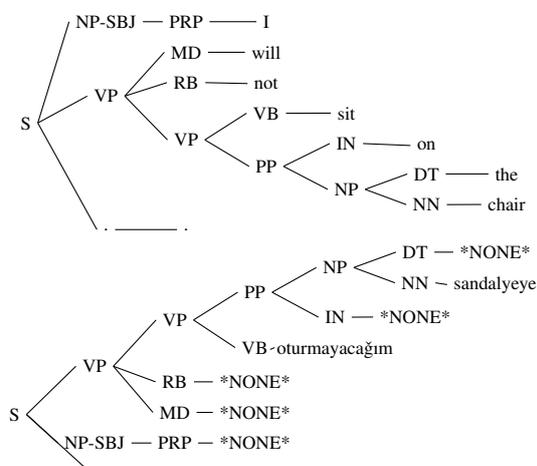


Figure 3: Original and translated trees, sandalye-ye otur-ma-yacağ-ım chair-DAT sit-NEG-FUT-1SG

5.2 The determiner “the”

There is no definite article in Turkish corresponding to “the”. Depending on the context, “the” is translated either as *NONE* or one of the demonstrative adjectives in Turkish, corresponding to “this” and “that” in English. See Figure 3.

5.3 Case markers

Turkish, being a fairly scrambling language, uses case markers to denote the syntactic functions of nouns and noun groups. For example, accusative case may be used to mark the direct object of a transitive verb and locative case may be used to mark the head of a prepositional phrase. In translation from English to Turkish, the prepositions are usually replaced with *NONE* and their corresponding case is attached to the nominal head of the phrase. See Figure 4.

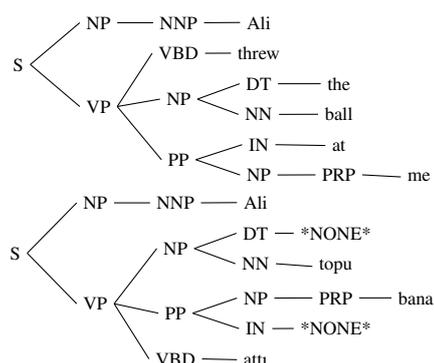


Figure 4: Original and translated trees, Ali top-u ban-a at-tı Ali ball-ACC me-DAT throw-PAST-3SG

5.4 Plural in nouns and verb inflection

Number agreement between the verb in the predicate and the subject is somewhat loose in Turkish. We preserved this freedom in translation and chose the number inflection that sounds more natural. Also, plural nouns under NNS tag in the English tree are sometimes translated as singular. In those cases, we kept the original POS tag NNS intact but used the singular gloss. See Figure 5.

5.5 Tense ambiguity

It is in general not possible to find an exact mapping among the tense classes in a pair of languages. When translating the trees, we mapped the English verb tenses to their closest semantic classes in Turkish while trying to keep the overall flow of the Turkish sentence natural. In many cases, we mapped the perfective tense in English to the past tense in Turkish. Similarly, we sometimes mapped the present tense to present continuous. See Figure 5.

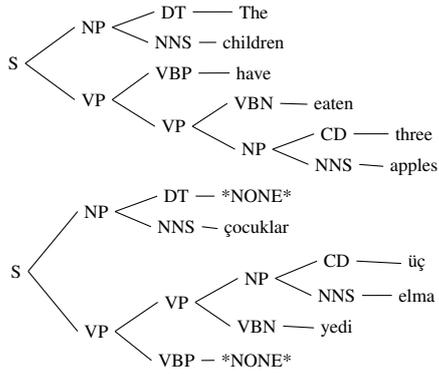


Figure 5: Original and translated trees,
Çocuk-lar üç elma ye-di
Child-PL three apple eat-PAST-3SG

5.6 WH- Questions

Question sentences require special attention during transformation. As opposed to movement in English question sentences, any constituent in Turkish can be questioned by replacing it with an inflected question word. In the Penn Treebank II annotation, the movement leaves a trace and is associated with wh- constituent with a numeric marker. For example, “WHNP-17” and “*T*-17” are associated.

When we translate the tree for a question sentence, we replace the wh- constituent with *NONE* and replace its trace with the appropriate question pronoun in Turkish. See Figure 6.

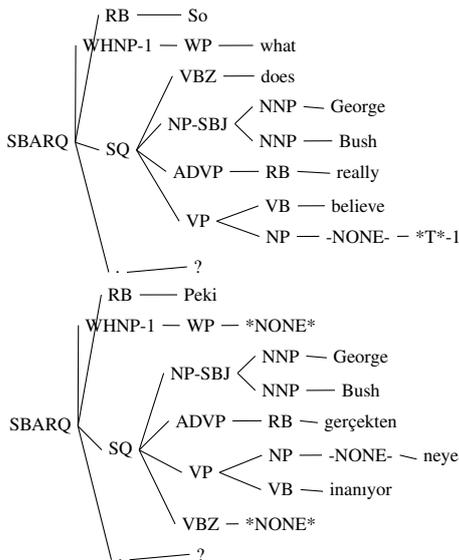


Figure 6: Original and translated trees,
Peki George Bush gerçekten ne-ye inan-ıyor?
So George Bush really what-DAT
believe-PRES-3SG?

5.7 Miscellany

In the translation of nominal clauses, the copula marker “-dIr” corresponding to verb “be” is often dropped.

The proper nouns are translated with their common Turkish gloss if there is one. So, “London” becomes “Londra”.

Subordinating conjunctions, marked as “IN” in English sentences, are transformed to *NONE* and the appropriate participle morpheme is appended to the stem in the Turkish translation.

A multiword expression may correspond to a single English word. Conversely, more than one words in English may correspond to a single word in Turkish. In the first case, we use the multiword expression as the gloss. In the latter case, we replace some English words with *NONE*.

6 Conclusion

Parallel treebank construction efforts increased significantly in the recent years. Many parallel treebanks are produced to build statistically strong language models for different languages. In this study, we report our preliminary efforts to build such a parallel corpus for Turkish-English pair. We translated and transformed a subset of parse trees of Penn Treebank to Turkish. We cover more than 50% of all sentences with a maximum length of 15-words including punctuation.

This work constitutes the preliminary step of parallel treebank generation. As a next step, we will focus on morphological analysis and disambiguation of Turkish words. After determining the correct morphological analysis of Turkish words, we will use the parts of these analyses to replace the leaf nodes that we intentionally left as “*NONE*”. As a future work, we plan to expand the dataset to include all Penn Treebank sentences.

References

Lars Ahrenberg. 2007. Lines: An english-swedish parallel treebank.

Nart B. Atalay, Kemal Oflazer, and Bilge Say. 2003. The annotation process in the Turkish treebank. In *4th International Workshop on Linguistically Interpreted Corpora*.

Ruken Cakici. 2005. Automatic induction of a ccg grammar for Turkish. In *ACL Student Research Workshop*.

- Ozlem Cetinoglu and Kemal Oflazer. 2006. Morphology-syntax interface for Turkish lfg. In *Computational Linguistics and Annual Meeting of the Association*.
- Ozlem Cetinoglu and Kemal Oflazer. 2009. Integrating derivational morphology into syntax. In *Recent Advances in Natural Language Processing V*.
- Martin Cmejrek, Jan Haji, and Vladislav Kubo. 2004. Prague czech-english dependency treebank: Syntactically annotated resources for machine translation. In *In Proceedings of EAMT 10th Annual Conference*, page 04.
- Lea Cyrus, Hendrik Feddes, and Frank Schumacher. 2003. FuSe – a multi-layered parallel treebank. In Joakim Nivre and Erhard Hinrichs, editors, *Proceedings of the Second Workshop on Treebanks and Linguistic Theories, 14–15 November 2003, Växjö, Sweden (TLT 2003)*, volume 9 of *Mathematical Modelling in Physics, Engineering and Cognitive Sciences*, pages 213–216, Växjö. Växjö University Press.
- Ilknur D. El-Kahlout. 2009. Statistical machine translation from english to turkish (ph.d. thesis).
- Gulsen Eryigit and Kemal Oflazer. 2006. Statistical dependency parsing for Turkish. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.
- Gulsen Eryigit, Esref Adali, and Kemal Oflazer. 2006. Türkçe cümlelerin kural tabanlı bağılılık analizi. In *15th Turkish Symposium on Artificial Intelligence and Neural Networks*.
- Gulsen Eryigit, Joakim Nivre, and Kemal Oflazer. 2008. Dependency parsing of Turkish. *Computational Linguistics*.
- Sofia Gustafson-Capkova, Yvonne Samuelsson, and Martin Volk. 2007. Smultron (version 1.0) - the stockholm multilingual parallel treebank. an english-german-swedish parallel treebank with sub-sentential alignments.
- Philipp Koehn. 2002. Europarl: A multilingual corpus for evaluation of machine translation.
- J. Kornfilt. 1997. *Turkish*. Routledge.
- Beáta Megyesi, Bengt Dahlqvist, Eva Pettersson, and Joakim Nivre. 2008. Swedish-turkish parallel treebank. In *LREC*.
- Beáta Megyesi, Bengt Dahlqvist, Éva Á. Csató, and Joakim Nivre. 2010. The english-swedish-turkish parallel treebank. In *LREC*.
- Joakim Nivre, Johan Hall, and Jens Nilsson. 2006a. Maltparser: A data-driven parser-generator for dependency parsing. In *In Proc. of LREC-2006*, pages 2216–2219.
- Joakim Nivre, Jens Nilsson, and Johan Hall. 2006b. Talbanken05: A swedish treebank with phrase structure and dependency annotation. In *In Proceedings of the fifth international conference on Language Resources and Evaluation (LREC2006)*, pages 24–26.
- S. Riedel, Ruket Cakici, and I. Meza-Ruiz. 2006. Multi-lingual dependency parsing with incremental integer linear programming.
- Ruket and Jason Baldridge. 2006. Projective and non-projective Turkish parsing. In *Fifth International Workshop on Treebanks and Linguistic Theories*.
- Sebastian Sulger, Miriam Butt, Tracy Holloway King, Paul Meurer, Tibor Laczkó, György Rákosi, Cheikh M. Bamba Dione, Helge Dyvik, Victoria Rosén, Koenraad De Smedt, Agnieszka Patejuk, Özlem Çetinoglu, I Wayan Arka, and Meladel Mistica. 2013. Pargrambank: The pargram parallel treebank. In *ACL (1)*, pages 550–560.
- Reyyan Yeniterzi and Kemal Oflazer. 2010. Syntax-to-morphology mapping in factored phrase-based statistical machine translation from english to turkish. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, pages 454–464, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Deniz Yuret. 2006. Dependency parsing as a classification problem. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*.

Improved Typesetting Models for Historical OCR

Taylor Berg-Kirkpatrick Dan Klein

Computer Science Division

University of California, Berkeley

{tberg, klein}@cs.berkeley.edu

Abstract

We present richer typesetting models that extend the unsupervised historical document recognition system of Berg-Kirkpatrick et al. (2013). The first model breaks the independence assumption between vertical offsets of neighboring glyphs and, in experiments, substantially decreases transcription error rates. The second model simultaneously learns multiple font styles and, as a result, is able to accurately track italic and non-italic portions of documents. Richer models complicate inference so we present a new, streamlined procedure that is over 25x faster than the method used by Berg-Kirkpatrick et al. (2013). Our final system achieves a relative word error reduction of 22% compared to state-of-the-art results on a dataset of historical newspapers.

1 Introduction

Modern OCR systems perform poorly on historical documents from the printing-press era, often yielding error rates that are too high for downstream research projects (Arlitsch and Herbert, 2004; Shoemaker, 2005; Holley, 2010). The two primary reasons that historical documents present difficulty for automatic systems are (1) the typesetting process used to produce such documents was extremely noisy and (2) the fonts used in the documents are unknown. Berg-Kirkpatrick et al. (2013) proposed a system for historical OCR that generatively models the noisy typesetting process of printing-press era documents and learns the font for each input document in an unsupervised fashion. Their system achieves state-of-the-art results on the task of historical document recognition.

We take the system of Berg-Kirkpatrick et al. (2013) as a starting point and consider extensions

of the typesetting model that address two shortcomings of their model: (1) their layout model assumes that baseline offset noise is independent for each glyph and (2) their font model assumes a single font is used in every document. Both of these assumptions are untrue in many historical datasets.

The baseline of the text in printing-press era documents is not rigid as in modern documents but rather drifts up and down noisily (see Figure 2). In practice, the vertical offsets of character glyphs change gradually along a line. This means the vertical offsets of neighboring glyphs are correlated, a relationship that is not captured by the original model. In our first extension, we let the vertical offsets of character glyphs be generated from a Markov chain, penalizing large changes in offset. We find that this extension decreases transcription error rates. Our system achieves a relative word error reduction of 22% compared to the state-of-the-art original model on a test set of historical newspapers (see Section 4.1), and a 11% relative reduction on a test set of historical court proceedings.

Multiple font styles are also frequently used in printing-press era documents; the most common scenario is for a basic font style to co-occur with an italic variant. For example, it is common for proper nouns and quotations to be italicized in the Old Bailey corpus (Shoemaker, 2005). In our second extension, we incorporate a Markov chain over font styles, extending the original model so that it is capable of simultaneously learning italic and non-italic fonts within a single document. In experiments, this model is able to detect which words are italicized with 93% precision at 74% recall in a test set of historical court proceedings (see Section 4.2).

These richer models that we propose do increase the state space and therefore make inference more costly. To remedy this, we streamline inference by replacing the coarse-to-fine inference scheme of Berg-Kirkpatrick et al. (2013)

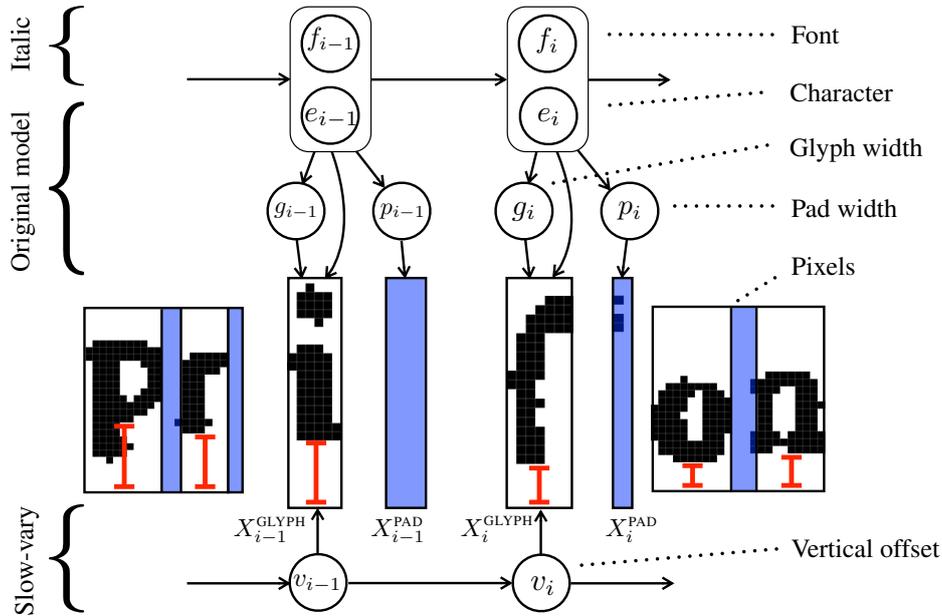


Figure 1: See Section 2 for a description of the generative process. We consider an extension of Berg-Kirkpatrick et al. (2013) that generates v_i conditioned on the previous vertical offset v_{i-1} (labeled Slow-vary) and an extension that generates a sequence of font styles f_i (labeled Italic).

with a forward-cost-augmented beaming scheme. Our method is over 25x faster on a typical document, yet actually yields *improved* transcriptions.

2 Model

We first describe the generative model used by the ‘Ocular’ historical OCR system of Berg-Kirkpatrick et al. (2013)¹ and then describe our extensions. The graphical model corresponding to their basic generative process for a single line of text is diagrammed in Figure 1. A Kneser-Ney (Kneser and Ney, 1995) character 6-gram language model generates a sequence of characters $E = (e_1, e_2, \dots, e_n)$. For each character index i , a glyph box width g_i and a pad box width p_i are generated, conditioned on the character e_i . g_i specifies the width of the bounding box that will eventually house the pixels of the glyph for character e_i . p_i specifies the width of a padding box which contains the horizontal space before the next character begins. Next, a vertical offset v_i is generated for the glyph corresponding to character e_i . v_i allows the model to capture variance in the baseline of the text in the document. We will later let v_i depend on v_{i-1} , as depicted in Figure 1, but in the baseline

¹The model we describe and extend has two minor differences from the one described by Berg-Kirkpatrick et al. (2013). While Berg-Kirkpatrick et al. (2013) generate two pad boxes for each character token, one to the left and one to the right, we only generate one pad box, always to the right. Additionally, Berg-Kirkpatrick et al. (2013) do not carry over the language model context between lines, while we do.

system they are independent. Finally, the pixels in the i th glyph bounding box X_i^{GLYPH} are generated conditioned on the character e_i , width g_i , and vertical offset v_i , and the pixels in the i th pad bounding box X_i^{PAD} are generated conditioned on the width p_i . We refer the reader to Berg-Kirkpatrick et al. (2013) for the details of the pixel generation process. We have omitted the token-level inking random variables for the purpose of brevity. These can be treated as part of the pixel generation process.

Let X denote the matrix of pixels for the entire line, $V = (v_1, \dots, v_n)$, $P = (p_1, \dots, p_n)$, and $G = (g_1, \dots, g_n)$. The joint distribution is written:

$$\begin{aligned}
 P(X, V, P, G, E) = & \\
 & P(E) && \text{[Language model]} \\
 & \cdot \prod_{i=1}^n P(g_i | e_i; \Phi) && \text{[Glyph widths]} \\
 & \cdot \prod_{i=1}^n P(p_i | e_i; \Phi) && \text{[Pad widths]} \\
 & \cdot \prod_{i=1}^n P(v_i) && \text{[Vertical offsets]} \\
 & \cdot \prod_{i=1}^n P(X_i^{PAD} | p_i) && \text{[Pad pixels]} \\
 & \cdot \prod_{i=1}^n P(X_i^{GLYPH} | v_i, g_i, e_i; \Phi) && \text{[Glyph pixels]}
 \end{aligned}$$

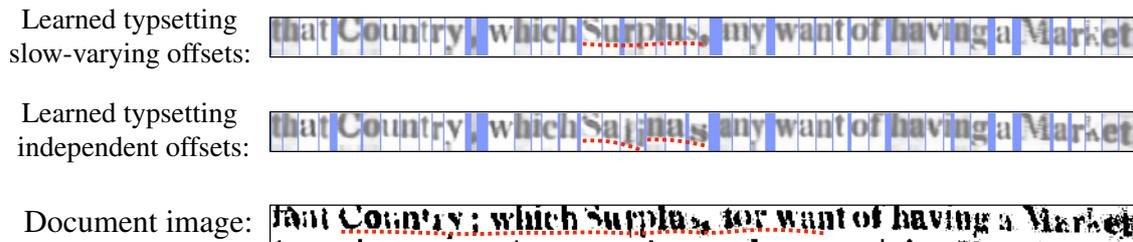


Figure 2: The first line depicts the Viterbi typesetting layout predicted by the OCULAR-BEAM-SV model. The second line depicts the same, but for the OCULAR-BEAM model. Pad boxes are shown in blue. Glyphs boxes are shown in white and display the Bernoulli template probabilities used to generate the observed pixels. The third line shows the corresponding portion of the input image.

The font is parameterized by the vector Φ which governs the shapes of glyphs and the distributions over box widths. Φ is learned in an unsupervised fashion. Document recognition is accomplished via Viterbi decoding over the character random variables e_i .

2.1 Slow-varying Offsets

The original model generates the vertical offsets v_i independently, and therefore cannot model how neighboring offsets are correlated. This correlation is actually strong in printing-press era documents. The baseline of the text wanders in the input image for two reasons: (1) the physical groove along which character templates were set was uneven and (2) the original document was imaged in a way that produced distortion. Both these underlying causes are likely to yield baselines that wander slowly up and down across a document. We refer to this behavior of vertical offsets as slow-varying, and extend the model to capture it.

In our first extension, we augment the model by incorporating a Markov chain over the vertical offset random variables v_i , as depicted in Figure 1. Specifically, v_i is generated from a discretized Gaussian centered at v_{i-1} :

$$P(v_i|v_{i-1}) \propto \exp\left(-\frac{(v_i - v_{i-1})^2}{2\sigma^2}\right)$$

This means that the if v_i differs substantially from v_{i-1} , a large penalty is incurred. As a result, the model should prefer sequences of v_i that vary slowly. In experiments, we set $\sigma^2 = 0.05$.

2.2 Italic Font Styles

Many of the documents in the Old Bailey corpus contain both italic and non-italic font styles (Shoemaker, 2005). The way that italic fonts are used depends on the year the document was printed, but generally italics are reserved for proper nouns,

quotations, and sentences that have a special role (e.g. the final judgment made in a court case). The switch between font styles almost always occurs at space characters.

Our second extension of the typesetting model deals with both italic and non-italic font styles. We augment the model with a Markov chain over font styles f_i , as depicted in Figure 1. Each font style token f_i takes on a value in $\{\text{ITALIC}, \text{NON-ITALIC}\}$ and is generated conditioned on the previous font style f_{i-1} and the current character token e_i . Specifically, after generating a character token that is not a space, the language model deterministically generates the last font used. If the language model generates a space character token, the decision of whether to switch font styles is drawn from a Bernoulli distribution. This ensures that the font style only changes at space characters.

The font parameters Φ are extended to contain entries for the italic versions of all characters. This means the shapes and widths of italic glyphs can be learned separately from non-italic ones. Like Berg-Kirkpatrick et al. (2013), we initialize the font parameters from mixtures of modern fonts, using mixtures of modern italic font styles for italic characters.

3 Streamlined Inference

Inference in our extended typesetting models is costly because the state space is large; we propose an new inference procedure that is fast and simple.

Berg-Kirkpatrick et al. (2013) used EM to learn the font parameters Φ , and therefore required expected sufficient statistics (indicators on (e_i, g_i, v_i) tuples), which they computed using coarse-to-fine inference (Petrov et al., 2008; Zhang and Gildea, 2008) with a semi-Markov dynamic program (Levinson, 1986). This approach is effec-

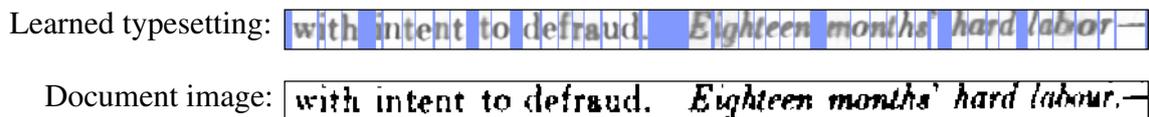


Figure 3: This first line depicts the Viterbi typesetting layout predicted by the OCULAR-BEAM-IT model. Pad boxes are shown in blue. Glyphs boxes are shown in white and display the Bernoulli template probabilities used to generate the observed pixels. The second line shows the corresponding portion of the input image.

tive, but slow. For example, while transcribing a typical document consisting of 30 lines of text, their system spends 63 minutes computing expected sufficient statistics and decoding when run on a 4.5GHz 4-core CPU.

We instead use hard counts of the sufficient statistics for learning (i.e. perform hard-EM). As a result, we are free to use inference procedures that are specialized for Viterbi computation. Specifically, we use beam-search with estimated forward costs. Because the model is semi-Markov, our beam-search procedure is very similar the one used by Pharaoh (Koehn, 2004) for phrase-based machine translation, only without a distortion model. We use a beam of size 20, and estimate forward costs using a character bigram language model. On the machine mentioned above, transcribing the same document, our simplified system that uses hard-EM and beam-search spends only 2.4 minutes computing sufficient statistics and decoding. This represents a 26x speedup.

4 Results

We ran experiments with four different systems. The first is our baseline, the system presented by Berg-Kirkpatrick et al. (2013), which we refer to as OCULAR. The second system uses the original model, but uses beam-search for inference. We refer to this system as OCULAR-BEAM. The final two systems use beam-search for inference, but use extended models: OCULAR-BEAM-SV uses the slow-varying vertical offset extension described in Section 2.1 and OCULAR-BEAM-IT uses the italic font extension described in Section 2.2.

We evaluate on two different test sets of historical documents. The first test set is called Trove, and is used by Berg-Kirkpatrick et al. (2013) for evaluation. Trove consists of 10 documents that were printed between 1803 and 1954, each consisting of 30 lines, all taken from a collection of historical Australian newspapers hosted by the National Library of Australia (Holley, 2010). The second test set, called Old Bailey, consists of 20

documents that were printed between 1716 and 1906, each consisting of 30 lines, all taken from the proceedings of the Old Bailey Courthouse in London (Shoemaker, 2005).² Following Berg-Kirkpatrick et al. (2013), we train the language model using 36 millions words from the New York Times portion of the Gigaword corpus (Graff et al., 2007).³

4.1 Document Recognition Performance

We evaluate predicted transcriptions using both character error rate (CER) and word error rate (WER). CER is the edit distance between the guessed transcription and the gold transcription, divided by the number of characters in the gold transcription. WER is computed in the same way, but words are treated as tokens instead of characters.

First we compare the baseline, OCULAR, to our system with simplified inference, OCULAR-BEAM. To our surprise, we found that OCULAR-BEAM produced better transcriptions than OCULAR. On Trove, OCULAR achieved a WER of 33.0 while OCULAR-BEAM achieved a WER of 30.7. On Old Bailey, OCULAR achieved a WER of 30.8 while OCULAR-BEAM achieved a WER of 28.8. These results are shown in Table 1, where we also report the performance of Google Tesseract (Smith, 2007) and ABBYY FineReader, a state-of-the-art commercial system, on the Trove test set (taken from Berg-Kirkpatrick et al. (2013)).

Next, we evaluate our slow-varying vertical offset model. OCULAR-BEAM-SV out-performs OCULAR-BEAM on both test sets. On Trove, OCULAR-BEAM-SV achieved a WER of 25.6, and on Old Bailey, OCULAR-BEAM-SV achieved a WER of 27.5. Overall, compared to our baseline

²Old Bailey is comparable to the the second test set used by Berg-Kirkpatrick et al. (2013) since it is derived from the same collection and covers a similar time span, but it consists of different documents.

³This means the language model is out-of-domain on both test sets. Berg-Kirkpatrick et al. (2013) also consider a perfectly in-domain language model, though this setting is somewhat unrealistic.

system, OCULAR-BEAM-SV achieved a relative reduction in WER of 22% on Trove and 11% on Old Bailey.

By looking at the predicted typesetting layouts we can make a qualitative comparison between the vertical offsets predicted by OCULAR-BEAM and OCULAR-BEAM-SV. Figure 2 shows representations of the Viterbi estimates of the typesetting random variables predicted by the models on a portion of an example document. The first line is the typesetting layout predicted by OCULAR-BEAM-SV and the second line is same, but for OCULAR-BEAM. The locations of padding boxes are depicted in blue. The white glyph bounding boxes reveal the values of the Bernoulli template probabilities used to generate the observed pixels. The Bernoulli templates are produced from type-level font parameters, but are modulated by token-level widths g_i and vertical offsets v_i (and inking random variables, whose description we have omitted for brevity). The predicted vertical offsets are visible in the shifted baselines of the template probabilities. The third line shows the corresponding portion of the input image. In this example, the text baseline predicted by OCULAR-BEAM-SV is contiguous, while the one predicted by OCULAR-BEAM is not. Given how OCULAR-BEAM-SV was designed, this meets our expectations. The text baseline predicted by OCULAR-BEAM has a discontinuity in the middle of its prediction for the gold word *Surplus*. In contrast, the vertical offsets predicted by OCULAR-BEAM-SV at this location vary smoothly and more accurately match the true text baseline in the input image.

4.2 Font Detection Performance

We ran experiments with the italic font style model, OCULAR-BEAM-IT, on the Old Bailey test set (italics are infrequent in Trove). We evaluated the learned styles by measuring how accurately OCULAR-BEAM-IT was able to distinguish between italic and non-italic styles. Specifically, we computed the precision and recall for the system’s predictions about which words were italicized. We found that, across the entire Old Bailey test set, OCULAR-BEAM-IT was able to detect which words were italicized with 93% precision at 74% recall, suggesting that the system did successfully learn both italic and non-italic styles.⁴

⁴While it seems plausible that learning italics could also improve transcription accuracy, we found that OCULAR-

System	CER	WER
Trove		
Google Tesseract	37.5	59.3
ABBYY FineReader	22.9	49.2
OCULAR (baseline)	14.9	33.0
OCULAR-BEAM	12.9	30.7
OCULAR-BEAM-SV	11.2	25.6
Old Bailey		
OCULAR (baseline)	14.9	30.8
OCULAR-BEAM	10.9	28.8
OCULAR-BEAM-SV	10.3	27.5

Table 1: We evaluate the output of each system on two test sets: Trove, a collection of historical newspapers, and Old Bailey, a collection of historical court proceedings. We report character error rate (CER) and word error rate (WER), macro-averaged across documents.

We can look at the typesetting layout predicted by OCULAR-BEAM-IT to gain insight into what has been learned by the model. The first line of Figure 3 shows the typesetting layout predicted by the OCULAR-BEAM-IT model for a line of a document image that contains italics. The second line of Figure 3 displays the corresponding portion of the input document image. From this example, it appears that the model has effectively learned separate glyph shapes for italic and non-italic versions of certain characters. For example, compare the template probabilities used to generate the *d*’s in *defraud* to the template probabilities used to generate the *d* in *hard*.

5 Conclusion

We began with an efficient simplification of the state-of-the-art historical OCR system of Berg-Kirkpatrick et al. (2013) and demonstrated two extensions to its underlying model. We saw an improvement in transcription quality as a result of removing a harmful independence assumption. This suggests that it may be worthwhile to consider still further extensions of the model, designed to more faithfully reflect the generative process that produced the input documents.

Acknowledgments

This work was supported by Grant IIS-1018733 from the National Science Foundation and also a National Science Foundation fellowship to the first author.

BEAM-IT actually performed slightly worse than OCULAR-BEAM. This negative result is possibly due to the extra difficulty of learning a larger number of font parameters.

References

- Kenning Arlitsch and John Herbert. 2004. Microfilm, paper, and OCR: Issues in newspaper digitization. the Utah digital newspapers program. *Microform & Imaging Review*.
- Taylor Berg-Kirkpatrick, Greg Durrett, and Dan Klein. 2013. Unsupervised transcription of historical documents. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*.
- David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2007. English Gigaword third edition. Linguistic Data Consortium, Catalog Number LDC2007T07.
- Rose Holley. 2010. Trove: Innovation in access to information in Australia. *Ariadne*.
- Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*.
- Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Machine translation: From real users to research*, pages 115–124. Springer.
- Stephen Levinson. 1986. Continuously variable duration hidden Markov models for automatic speech recognition. *Computer Speech & Language*.
- Slav Petrov, Aria Haghghi, and Dan Klein. 2008. Coarse-to-fine syntactic machine translation using language projections. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*.
- Robert Shoemaker. 2005. Digital London: Creating a searchable web of interlinked sources on eighteenth century London. *Electronic Library and Information Systems*.
- Ray Smith. 2007. An overview of the Tesseract OCR engine. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition*.
- Hao Zhang and Daniel Gildea. 2008. Efficient multi-pass decoding for synchronous context free grammars. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*.

Robust Logistic Regression using Shift Parameters

Julie Tibshirani and Christopher D. Manning

Stanford University

Stanford, CA 94305, USA

{jtibs, manning}@cs.stanford.edu

Abstract

Annotation errors can significantly hurt classifier performance, yet datasets are only growing noisier with the increased use of Amazon Mechanical Turk and techniques like distant supervision that automatically generate labels. In this paper, we present a robust extension of logistic regression that incorporates the possibility of mislabelling directly into the objective. This model can be trained through nearly the same means as logistic regression, and retains its efficiency on high-dimensional datasets. We conduct experiments on named entity recognition data and find that our approach can provide a significant improvement over the standard model when annotation errors are present.

1 Introduction

Almost any large dataset has annotation errors, especially those complex, nuanced datasets commonly used in natural language processing. Low-quality annotations have become even more common in recent years with the rise of Amazon Mechanical Turk, as well as methods like distant supervision and co-training that involve automatically generating training data.

Although small amounts of noise may not be detrimental, in some applications the level can be high: upon manually inspecting a relation extraction corpus commonly used in distant supervision, Riedel et al. (2010) report a 31% false positive rate. In cases like these, annotation errors have frequently been observed to hurt performance. Dingare et al. (2005), for example, conduct error analysis on a system to extract relations from biomedical text, and observe that over half of the system's errors could be attributed to inconsistencies in how the data was annotated. Similarly, in a case study on co-training for natural lan-

guage tasks, Pierce and Cardie (2001) find that the degradation in data quality from automatic labelling prevents these systems from performing comparably to their fully-supervised counterparts.

In this work we argue that incorrect examples should be explicitly modelled during training, and present a simple extension of logistic regression that incorporates the possibility of mislabelling directly into the objective. Following a technique from robust statistics, our model introduces sparse 'shift parameters' to allow datapoints to slide along the sigmoid, changing class if appropriate. It has a convex objective, is well-suited to high-dimensional data, and can be efficiently trained with minimal changes to the logistic regression pipeline.

In experiments on a large, noisy NER dataset, we find that this method can provide an improvement over standard logistic regression when annotation errors are present. The model also provides a means to identify which examples were mislabelled: through experiments on biological data, we demonstrate how our method can be used to accurately identify annotation errors. This robust extension of logistic regression shows particular promise for NLP applications: it helps account for incorrect labels, while remaining efficient on large, high-dimensional datasets.

2 Related Work

Much of the previous work on dealing with annotation errors centers around filtering the data before training. Brodley and Friedl (1999) introduce what is perhaps the simplest form of supervised filtering: they train various classifiers, then record their predictions on a different part of the train set and eliminate contentious examples. Sculley and Cormack (2008) apply this approach to spam filtering with noisy user feedback.

One obvious issue with these methods is that the noise-detecting classifiers are themselves trained

on noisy labels. Unsupervised filtering tries to avoid this problem by clustering training instances based solely on their features, then using the clusters to detect labelling anomalies (Rebbapragada et al., 2009). Recently, Intxaurreondo et al. (2013) applied this approach to distantly-supervised relation extraction, using heuristics such as the number of mentions per tuple to eliminate suspicious examples.

Unsupervised filtering, however, relies on the perhaps unwarranted assumption that examples with the same label lie close together in feature space. Moreover filtering techniques in general may not be well-justified: if a training example does not fit closely with the current model, it is not necessarily mislabelled. It may represent an important exception that would improve the overall fit, or appear unusual simply because we have made poor modelling assumptions.

Perhaps the most promising approaches are those that directly model annotation errors, handling mislabelled examples as they train. This way, there is an active trade-off between fitting the model and identifying suspected errors. Bootkrajang and Kaban (2012) present an extension of logistic regression that models annotation errors through flipping probabilities. While intuitive, this approach has shortcomings of its own: the objective function is nonconvex and the authors note that local optima are an issue, and the model can be difficult to fit when there are many more features than training examples.

There is a growing body of literature on learning from several annotators, each of whom may be inaccurate (Bachrach et al., 2012; Raykar et al., 2009). It is important to note that we are considering a separate, and perhaps more general, problem: we have only one source of noisy labels, and the errors need not come from the human annotators, but could be introduced through contamination or automatic labelling.

The field of ‘robust statistics’ seeks to develop estimators that are not unduly affected by deviations from the model assumptions (Huber and Ronchetti, 2009). Since mislabelled points are one type of outlier, this goal is naturally related to our interest in dealing with noisy data, and it seems many of the existing techniques would be relevant. A common strategy is to use a modified loss function that gives less influence to points far from the boundary, and several models along

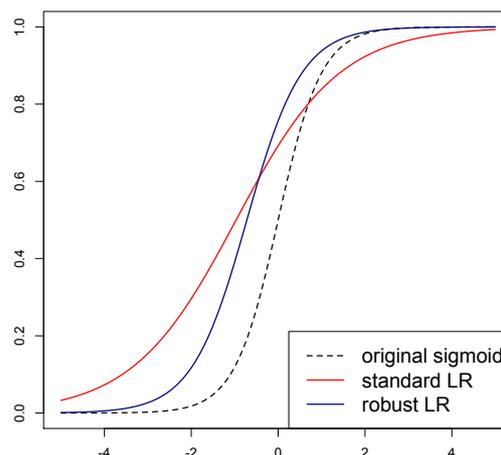


Figure 1: Fit resulting from a standard vs. robust model, where data is generated from the dashed sigmoid and negative labels flipped with probability 0.2.

these lines have been proposed (Ding and Vishwanathan., 2010; Masnadi-Shirazi et al., 2010). Unfortunately these approaches require optimizing nonstandard, often nonconvex objectives, and fail to give insight into which datapoints are mislabelled.

In a recent advance, She and Owen (2011) demonstrate that introducing a regularized ‘shift parameter’ per datapoint can help increase the robustness of linear regression. Candès et al. (2009) propose a similar approach for principal component analysis, while Wright and Ma (2009) explore its effectiveness in sparse signal recovery. In this work we adapt the technique to logistic regression. To the best of our knowledge, we are the first to experiment with adding ‘shift parameters’ to logistic regression and demonstrate that the model is especially well-suited to the type of high-dimensional, noisy datasets commonly used in NLP.

3 Model

Recall that in binary logistic regression, the probability of an example x_i being positive is modeled as

$$g(\theta^T x_i) = \frac{1}{1 + e^{-\theta^T x_i}}.$$

For simplicity, we assume the intercept term has been folded into the weight vector θ , so $\theta \in \mathbb{R}^{m+1}$ where m is the number of features.

Following She and Owen (2011), we propose the following robust extension: for each datapoint $i = 1, \dots, n$, we introduce a real-valued shift pa-

parameter γ_i so that the sigmoid becomes

$$g(\theta^T x_i + \gamma_i) = \frac{1}{1 + e^{-\theta^T x_i - \gamma_i}}.$$

Since we believe that most examples are correctly labelled, we L_1 -regularize the shift parameters to encourage sparsity. Letting $y_i \in \{0, 1\}$ be the label for datapoint i and fixing $\lambda \geq 0$, our objective is now given by

$$l(\theta, \gamma) = \sum_{i=1}^n \left[y_i \log g(\theta^T x_i + \gamma_i) + (1 - y_i) \log (1 - g(\theta^T x_i + \gamma_i)) \right] - \lambda \sum_{i=1}^n |\gamma_i|. \quad (1)$$

These parameters γ_i let certain datapoints shift along the sigmoid, perhaps switching from one class to the other. If a datapoint i is correctly annotated, then we would expect its corresponding γ_i to be zero. If it actually belongs to the positive class but is labelled negative, then γ_i might be positive, and analogously for the other direction.

One way to interpret the model is that it allows the log-odds of select datapoints to be shifted. Compared to models based on label-flipping, where there is a global set of flipping probabilities, our method has the advantage of targeting each example individually.

It is worth noting that there is no difficulty in regularizing the θ parameters as well. For example, if we choose to use an L_1 penalty then our objective becomes

$$l(\theta, \gamma) = \sum_{i=1}^n \left[y_i \log g(\theta^T x_i + \gamma_i) + (1 - y_i) \log (1 - g(\theta^T x_i + \gamma_i)) \right] - \kappa \sum_{j=1}^m |\theta_j| - \lambda \sum_{i=1}^n |\gamma_i|. \quad (2)$$

Finally, it may seem concerning that we have introduced a new parameter for each datapoint. But in many applications the number of features already exceeds n , so with proper regularization, this increase is actually quite reasonable.

3.1 Training

Notice that adding these shift parameters is equivalent to introducing n features, where the i th new feature is 1 for datapoint i and 0 otherwise. With

this observation, we can simply modify the feature matrix and parameter vector and train the logistic model as usual. Specifically, we let $\theta' = (\theta_0, \dots, \theta_m, \gamma_1, \dots, \gamma_n)$ and $X' = [X | I_n]$ so that the objective (1) simplifies to

$$l(\theta') = \sum_{i=1}^n \left[y_i \log g(\theta'^T x'_i) + (1 - y_i) \log (1 - g(\theta'^T x'_i)) \right] - \lambda \sum_{j=m+1}^{m+n} |\theta'^{(j)}|.$$

Upon writing the objective in this way, we immediately see that it is convex, just as standard L_1 -penalized logistic regression is convex.

3.2 Testing

To obtain our final logistic model, we keep only the θ parameters. Predictions are then made as usual:

$$\mathbf{I}\{g(\hat{\theta}^T x) > 0.5\}.$$

3.3 Selecting Regularization Parameters

The parameter λ from equation (1) would normally be chosen through cross-validation, but our set-up is unusual in that the training set may contain errors, and even if we have a designated development set it is unlikely to be error-free. We found in simulations that the errors largely do not interfere in selecting λ , so in the experiments below we cross-validate as normal.

Notice that λ has a direct effect on the number of nonzero shifts γ and hence the suspected number of errors in the training set. So if we have information about the noise level, we can directly incorporate it into the selection procedure. For example, we may believe the training set has no more than 15% noise, and so would restrict the choice of λ during cross-validation to only those values where 15% or fewer of the estimated shift parameters are nonzero.

We now consider situations in which the θ parameters are regularized as well. Assume, for example, that we use L_1 -regularization as in equation (2), so that we now need to optimize over both κ and λ . We perform the following simple procedure:

1. Cross-validate using standard logistic regression to select κ .
2. Fix this value for κ , and cross-validate using the robust model to find the best choice of λ .

method	suspects identified										false positives
	T2	T30	T33	T36	T37	N8	N12	N34	N36		
Alon et al. (1999)											
Furey et al. (2000)		•	•	•		•		•	•		
Kadota et al. (2003)	•				•	•		•	•		T6, N2
Malossini et al. (2006)	•	•	•	•			•	•	•		T8, N2, N28, N29
Bootkrajang et al. (2012)	•	•	•	•			•	•	•		
Robust LR		•	•	•		•	•	•	•		

Table 1: Results of various error-identification methods on the colon cancer dataset. The first row lists the samples that are biologically confirmed to be suspicious, and each other row gives the output from an automatic detection method. Bootkrajang et al. report confidences, so we threshold at 0.5 to obtain these results.

4 Experiments

We conduct two sets of experiments to assess the effectiveness of the approach, in terms of both identifying mislabelled examples and producing accurate predictions.

4.1 Contaminated Data

Our first experiment is centered around a biological dataset with suspected labelling errors. Called the colon cancer dataset, it contains the expression levels of 2000 genes from 40 tumor and 22 normal tissues (Alon et al., 1999). There is evidence in the literature that certain tissue samples may have been cross-contaminated. In particular, 5 tumor and 4 normal samples should have their labels flipped.

In this experiment, we examine the model’s ability to identify mislabelled training examples. Because there are many more features than datapoints and it is likely that not all genes are relevant, we choose to place an L_1 penalty on θ .

Using `glmnet`, an R package for training regularized models (Friedman et al., 2009), we select κ and λ using cross-validation. Looking at the resulting values for γ , we find that only 7 of the shift parameters are nonzero and that each one corresponds to a suspicious datapoint. As further confirmation, the signs of the gammas correctly match the direction of the mislabelling. Compared to previous attempts to automatically detect errors in this dataset, our approach identifies at least as many suspicious examples but with no false positives. A detailed comparison is given in Table 1. Although Bootkrajang and Kaban (2012) are quite accurate, it is worth noting that due to its nonconvexity, their model needed to be trained 20 times to achieve these results.

4.2 Manually Annotated Data

We now consider the problem of *named entity recognition* (NER) to evaluate how our model performs in a large-scale prediction task. In traditional NER, the goal is to determine whether each word is a person, organization, location, or not a named entity (‘other’). Since our model is binary, we concentrate on the task of deciding whether a word is a person or not. (This task does not trivially reduce to finding the capitalized words, as the model must distinguish between people and other named entities like organizations).

For training, we use a large, noisy NER dataset collected by Jenny Finkel. The data was created by taking various Wikipedia articles and giving them to five Amazon Mechanical Turkers to annotate. Few to no quality controls were put in place, so that certain annotators produced very noisy labels. To construct the train set we chose a Turker who was about average in how much he disagreed with the majority vote, and used only his annotations. Negative examples are subsampled to bring the class ratio to a reasonable level, for a total of 200,000 negative and 24,002 positive examples. We find that in 0.4% of examples, the majority agreed they were negative but the chosen annotator marked them positive, and 7.5% were labelled positive by the majority but negative by the annotator. Note that we still include examples for which there was no majority consensus, so these noise estimates are quite conservative.

We evaluate on the English development test set from the CoNLL shared task (Tjong Kim Sang and Meulder, 2003). This data consists of news articles from the Reuters corpus, hand-annotated by researchers at the University of Antwerp.

We extract a set of features using Stanford’s NER pipeline (Finkel et al., 2005). This set was

model	precision	recall	F1
standard	76.99	85.87	81.19
flipping	76.62	86.28	81.17
robust	77.04	90.47	83.22

Table 2: Performance of standard vs. robust logistic regression in the Wikipedia NER experiment. The flipping model refers to the approach from Bootkrajang and Kaban (2012).

chosen for simplicity and is not highly engineered – it largely consists of lexical features such as the current word, the previous and next words in the sentence, as well as character n-grams and various word shape features. With a total of 393,633 features in the train set, we choose to use L_2 -regularization, so that our penalty now becomes

$$\frac{1}{2\sigma^2} \sum_{j=0}^m |\theta_j|^2 + \lambda \sum_{i=1}^n |\gamma_i|.$$

This choice is natural as L_2 is the most common form of regularization in NLP, and we wish to verify that our approach works for penalties besides L_1 .

The robust model is fit using Orthant-Wise Limited-Memory Quasi Newton (OWL-QN), a technique for optimizing an L_1 -penalized objective (Andrew and Gao, 2007). We tune both models through 5-fold cross-validation to obtain $\sigma^2 = 1.0$ and $\lambda = 0.1$. Note that from the way we cross-validate (first tuning σ using standard logistic regression, fixing this choice, then tuning λ) our procedure may give an unfair advantage to the baseline.

We also compare against the algorithm proposed in Bootkrajang and Kaban (2012), an extension of logistic regression mentioned in the section on prior work. This approach assumes that each example’s true label is flipped with a certain probability before being observed, and fits the resulting latent-variable model using EM.

The results of these experiments are shown in Table 2 as well as Figure 2. Robust logistic regression offers a noticeable improvement over the baseline, and this improvement holds at essentially all levels of precision and recall. Interestingly, because of the large dimension, the flipping model consistently learns that no labels have been flipped and thus does not show a substantial difference with standard logistic regression.

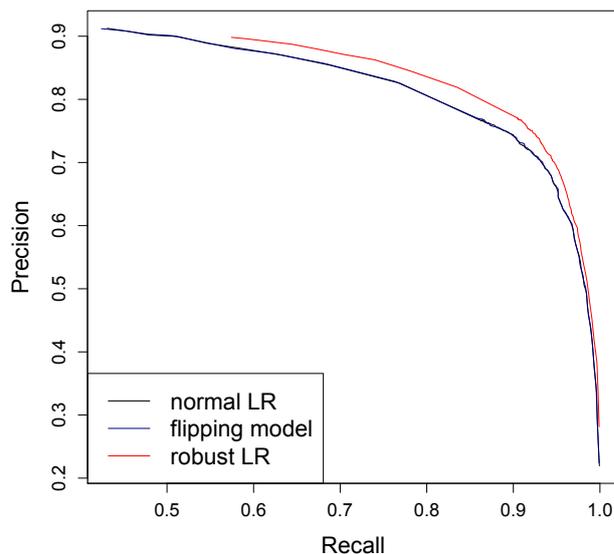


Figure 2: Precision-recall curve obtained from training on noisy Wikipedia data and testing on CoNLL. The flipping model refers to the approach from Bootkrajang and Kaban (2012).

5 Future Work

A natural direction for future work is to extend the model to a multi-class setting. One option is to introduce a γ for every class except the negative one, so that there are $n(c - 1)$ shift parameters in all. We could then apply a group lasso, with each group consisting of the γ for a particular datapoint (Meier et al., 2008). This way all of a datapoint’s shift parameters drop out together, which corresponds to the example being correctly labelled.

CRFs and other sequence models could also benefit from the addition of shift parameters. Since the extra variables can be neatly folded into the linear term, convexity is preserved and the model could essentially be trained as usual.

Acknowledgments

Stanford University gratefully acknowledges the support of the Defense Advanced Research Projects Agency (DARPA) Deep Exploration and Filtering of Text (DEFT) Program under Air Force Research Laboratory (AFRL) contract no. FA8750-13-2-0040. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government. We are especially grateful to Rob Tibshirani and Stefan Wager for their invaluable advice and encouragement.

References

- U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, A. J. Levine. 1999. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *National Academy of Sciences of the USA*.
- Galen Andrew and Jianfeng Gao. 2007. Scalable Training of L_1 -Regularized Log-Linear Models. *ICML*.
- Yoram Bachrach, Thore Graepel, Tom Minka, and John Guiver. 2012. How To Grade a Test Without Knowing the Answers: A Bayesian Graphical Model for Adaptive Crowdsourcing and Aptitude Testing. *arXiv preprint arXiv:1206.6386 (2012)*.
- Jakramate Bootkrajang and Ata Kaban. 2012. Label-noise Robust Logistic Regression and Its Applications. *ECML PKDD*.
- Carla E. Brodley and Mark A. Friedl. 1999. Identifying mislabeled Training Data. *JAIR*, 11, 131-167.
- Emmanuel J. Candes, Xiaodong Li, Yi Ma, John Wright. 2009. Robust Principal Component Analysis? *arXiv preprint arXiv:0912.3599, 2009*.
- Nan Ding and S. V. N. Vishwanathan. 2010. t-Logistic regression. *NIPS*.
- Shipra Dingare, Malvina Nissim, Jenny Finkel, Christopher Manning, and Claire Grover. 2005. A system for identifying named entities in biomedical text: How results from two evaluations reflect on both the system and the evaluations. *Comparative and Functional Genomics*. 6(1–2), 77-85.
- Jenny Rose Finkel, Trond Grenager, Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. *ACL*.
- Jerome Friedman, Trevor Hastie, Rob Tibshirani. 2009. Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of statistical software*, 33(1), 1.
- Terrence S. Furey, Nello Cristianini, Nigel Duffy, David W. Bednarski, Michel Schummer, David Haussler. 2000. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10), 906-914.
- Peter J. Huber and Elvezio M. Ronchetti. 2000. *Robust Statistics*. John Wiley & Sons, Inc., Hoboken, NJ.
- Ander Intxaurre, Mihai Surdeanu, Oier Lopez de Lacalle, and Eneko Agirre. 2013. Removing Noisy Mentions for Distant Supervision. *Congreso de la Sociedad Espaola para el Procesamiento del Lenguaje Natural*.
- Koji Kadota, Daisuke Tominaga, Yutaka Akiyama, Katsutoshi Takahashi. 2003. Detecting outlying samples in microarray data: A critical assessment of the effect of outliers on sample. *ChemBio Informatics Journal*, 3(1), 30-45.
- Andrea Malossini, Enrico Blanzieri, Raymond T. Ng. 2006. Detecting potential labeling errors in microarrays by data perturbation. *Bioinformatics*, 22(17), 2114-2121.
- Hamed Masnadi-Shirazi, Vijay Mahadevan, and Nuno Vasconcelos. 2010. On the design of robust classifiers for computer vision. *IEEE International Conference Computer Vision and Pattern Recognition*.
- Lukas Meier, Sara van de Geer, Peter Bühlmann. 2008. The group lasso for logistic regression. *Journal of the Royal Statistical Society*, 70(1), 53-71.
- David Pierce and Claire Cardie. 2001. Limitations of co-training for natural language learning from large datasets. *EMNLP*.
- Vikas Raykar, Shipeng Yu, Linda H. Zhao, Anna Jerebko, Charles Florin, Gerardo Hermosillo Valadez, Luca Bogoni, and Linda Moy. 2009. Supervised learning from multiple experts: whom to trust when everyone lies a bit. *ICML*.
- Umaa Rebbapragada, Lukas Mandrake, Kiri L. Wagstaff, Damhnait Gleeson, Rebecca Castano, Steve Chien, Carla E. Brodley. 2009. Improving Onboard Analysis of Hyperion Images by Filtering mislabelled Training Data Examples. *IEEE Aerospace Conference*.
- Sebastian Riedel, Limin Yao, Andrew McCallum. 2010. Modeling Relations and Their Mentions without Labelled Text. *ECML PKDD*.
- D. Sculley and Gordon V. Cormack. 2008. Filtering Email Spam in the Presence of Noisy User Feedback. *CEAS*.
- Yiyuan She and Art Owen. 2011. Outlier Detection Using Nonconvex Penalized Regression. *Journal of the American Statistical Association*, 106(494).
- Erik F. Tjong Kim Sang, Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. *CoNLL*.
- John Wright and Yi Ma. 2009. Dense Error Correction via l_1 -Minimization *IEEE Transactions on Information Theory*.

Faster Phrase-Based Decoding by Refining Feature State

Kenneth Heafield **Michael Kayser** **Christopher D. Manning**
Computer Science Department Stanford University, Stanford, CA, 94305
{heafield,mkayser,manning}@stanford.edu

Abstract

We contribute a faster decoding algorithm for phrase-based machine translation. Translation hypotheses keep track of state, such as context for the language model and coverage of words in the source sentence. Most features depend upon only part of the state, but traditional algorithms, including cube pruning, handle state atomically. For example, cube pruning will repeatedly query the language model with hypotheses that differ only in source coverage, despite the fact that source coverage is irrelevant to the language model. Our key contribution avoids this behavior by placing hypotheses into equivalence classes, masking the parts of state that matter least to the score. Moreover, we exploit shared words in hypotheses to iteratively refine language model scores rather than handling language model state atomically. Since our algorithm and cube pruning are both approximate, improvement can be used to increase speed or accuracy. When tuned to attain the same accuracy, our algorithm is 4.0–7.7 times as fast as the Moses decoder with cube pruning.

1 Introduction

Translation speed is critical to making suggestions as translators type, mining for parallel data by translating the web, and running on mobile devices without Internet connectivity. We contribute a fast decoding algorithm for phrase-based machine translation along with an implementation in a new open-source (LGPL) decoder available at <http://kheafield.com/code/>.

Phrase-based decoders (Koehn et al., 2007; Cer et al., 2010; Wuebker et al., 2012) keep track of several types of state with translation hypothe-

ses: coverage of the source sentence thus far, context for the language model, the last position for the distortion model, and anything else features need. Existing decoders handle state atomically: hypotheses that have exactly the same state can be recombined and efficiently handled via dynamic programming, but there is no special handling for partial agreement. Therefore, features are repeatedly consulted regarding hypotheses that differ only in ways irrelevant to their score, such as coverage of the source sentence. Our decoder bundles hypotheses into equivalence classes so that features can focus on the relevant parts of state.

We pay particular attention to the language model because it is responsible for much of the hypothesis state. As the decoder builds translations from left to right (Koehn, 2004), it records the last $N - 1$ words of each hypothesis so that they can be used as context to score the first $N - 1$ words of a phrase, where N is the order of the language model. Traditional decoders (Huang and Chiang, 2007) try thousands of combinations of hypotheses and phrases, hoping to find ones that the language model likes. Our algorithm instead discovers good combinations in a coarse-to-fine manner. The algorithm exploits the fact that hypotheses often share the same suffix and phrases often share the same prefix. These shared suffixes and prefixes allow the algorithm to coarsely reason over many combinations at once.

Our primary contribution is a new search algorithm that exploits the above observations, namely that state can be divided into pieces relevant to each feature and that language model state can be further subdivided. The primary claim is that our algorithm is faster and more accurate than the popular cube pruning algorithm.

2 Related Work

Our previous work (Heafield et al., 2013) developed language model state refinement for bottom-

up decoding in syntactic machine translation. In bottom-up decoding, hypotheses can be extended to the left or right, so hypotheses keep track of both their prefix and suffix. The present phrase-based setting is simpler because sentences are constructed from left to right, so prefix information is unnecessary. However, phrase-based translation implements reordering by allowing hypotheses that translate discontinuous words in the source sentence. There are exponentially many ways to cover the source sentence and hypotheses carry this information as additional state. A main contribution in this paper is efficiently ignoring coverage when evaluating the language model. In contrast, syntactic machine translation hypotheses correspond to contiguous spans in the source sentence, so in prior work we simply ran the search algorithm in every span.

Another improvement upon Heafield et al. (2013) is that we previously made no effort to exploit common words that appear in translation rules, which are analogous to phrases. In this work, we explicitly group target phrases by common prefixes, doing so directly in the phrase table.

Coarse-to-fine approaches (Petrov et al., 2008; Zhang and Gildea, 2008) invoke the decoder multiple times with increasingly detailed models, pruning after each pass. The key difference in our work is that, rather than refining models in lock step, we effectively refine the language model on demand for hypotheses that score well. Moreover, their work was performed in syntactic machine translation while we address issues specific to phrase-based translation.

Our baseline is cube pruning (Chiang, 2007; Huang and Chiang, 2007), which is both a way to organize search and an algorithm to search through cross products of sets. We adopt the same search organization (Section 3.1) but change how cross products are searched.

Chang and Collins (2011) developed an exact decoding algorithm based on Lagrangian relaxation. However, it has not been shown to tractably scale to 5-gram language models used by many modern translation systems.

3 Decoding

We begin by summarizing the high-level organization of phrase-based cube pruning (Koehn, 2004; Koehn et al., 2007; Huang and Chiang, 2007). Sections 3.2 and later show our contribution.

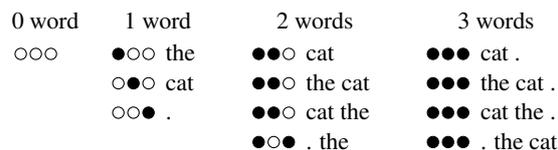


Figure 1: Stacks to translate the French “le chat .” into English. Filled circles indicate that the source word has been translated. A phrase translates “le chat” as simply “cat”, emphasizing that stacks are organized by the number of source words rather than the number of target words.

3.1 Search Organization

Phrase-based decoders construct hypotheses from left to right by appending phrases in the target language. The decoder organizes this search process using *stacks* (Figure 1). Stacks contain hypotheses that have translated the same number of source words. The zeroth stack contains one hypothesis with nothing translated. Subsequent stacks are built by extending hypotheses in preceding stacks. For example, the second stack contains hypotheses that translated two source words either separately or as a phrasal unit. Returning to Figure 1, the decoder can apply a phrase pair to translate “le chat” as “cat” or it can derive “the cat” by translating one word at a time; both appear in the second stack because they translate two source words. To generalize, the decoder populates the i th stack by pairing hypotheses in the $i - j$ th stack with target phrases that translate source phrases of length j . Hypotheses remember which source word they translated, as indicated by the filled circles.

The reordering limit prevents hypotheses from jumping around the source sentence too much and dramatically reduces the search space. Formally, the decoder cannot propose translations that would require jumping back more than R words in the source sentence, including multiple small jumps.

In practice, stacks are limited to k hypotheses, where k is set by the user. Small k is faster but may prune good hypotheses, while large k is slower but more thorough, thereby comprising a time-accuracy trade-off. The central question in this paper is how to select these k hypotheses.

Populating a stack boils down to two steps. First, the decoder matches hypotheses with source phrases subject to three constraints: the total source length matches the stack being populated, none of the source words has already been trans-

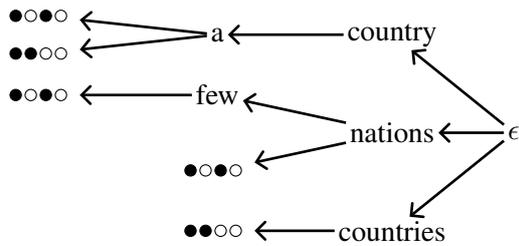


Figure 2: Hypothesis suffixes arranged into a trie. The leaves indicate source coverage and any other hypothesis state.

lated, and the reordering limit. Second, the decoder searches through these matches to select k high-scoring hypotheses for placement in the stack. We improve this second step.

The decoder provides our algorithm with pairs consisting of a hypothesis and a compatible source phrase. Each source phrase translates to multiple target phrases. The task is to grow these hypotheses by appending a target phrase, yielding new hypotheses. These new hypotheses will be placed into a stack of size k , so we are interested in selecting k new hypotheses that score highly.

Beam search (Lowerre, 1976; Koehn, 2004) tries every hypothesis with every compatible target phrase then selects the top k new hypotheses by score. This is wasteful because most hypotheses are discarded. Instead, we follow cube pruning (Chiang, 2007) in using a priority queue to generate k hypotheses. A key difference is that we generate these hypotheses iteratively.

3.2 Tries

For each source phrase, we collect the set of compatible hypotheses. We then place these hypotheses in a trie that emphasizes the suffix words because these matter most when appending a target phrase. Figure 2 shows an example. While it suffices to build this trie on the last $N - 1$ words that matter to the language model, Li and Khudanpur (2008) have identified cases where fewer words are necessary because the language model will back off. The leaves of the trie are complete hypotheses and reveal information irrelevant to the language model, such as coverage of the source sentence and the state of other features.

Each source phrase translates to a set of target phrases. Because these phrases will be appended to a hypothesis, the first few words matter the most to the language model. We therefore

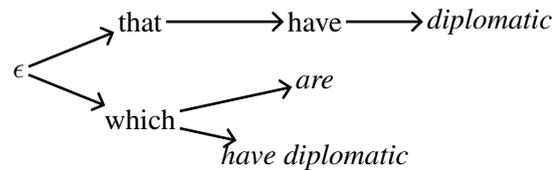


Figure 3: Target phrases arranged into a trie. Set in *italic*, leaves reveal parts of the phrase that are irrelevant to the language model.

arrange the target phrases into a prefix trie. An example is shown in Figure 3. Similar to the hypothesis trie, the depth may be shorter than $N - 1$ in cases where the language model will provably back off (Li and Khudanpur, 2008). The trie can also be short because the target phrase has fewer than $N - 1$ words. We currently store this trie data structure directly in the phrase table, though it could also be computed on demand to save memory. Empirically, our phrase table uses less RAM than Moses’s memory-based phrase table.

As an optimization, a trie reveals multiple words when there would otherwise be no branching. This allows the search algorithm to make decisions only when needed.

Following Heafield et al. (2013), leaves in the trie take the score of the underlying hypothesis or target phrase. Non-leaf nodes take the maximum score of their descendants. Children of a node are sorted by score.

3.3 Boundary Pairs

The idea is that the decoder reasons over pairs of nodes in the hypothesis and phrase tries before delving into detail. In this way, it can determine what the language model likes and, conversely, quickly discard combinations that the model does not like.

A boundary pair consists of a node in the hypothesis trie and a node in the target phrase trie. For example, the decoder starts at the root of each trie with the boundary pair (ϵ, ϵ) . The score of a boundary pair is the sum of the scores of the underlying trie nodes. However, once some words have been revealed, the decoder calls the language model to compute a score adjustment. For example, the boundary pair (country, that) has score adjustment

$$\log \frac{p(\text{that} \mid \text{country})}{p(\text{that})}$$

times the weight of the language model. This has the effect of cancelling out the estimate made

when the phrase was scored in isolation, replacing it with a more accurate estimate based on available context. These score adjustments are efficient to compute because the decoder retained a pointer to “that” in the language model’s data structure (Heafield et al., 2011).

3.4 Splitting

Refinement is the notion that the boundary pair (ϵ, ϵ) divides into several boundary pairs that reveal specific words from hypotheses or target phrases. The most straightforward way to do this is simply to split into all children of a trie node. Continuing the example from Figure 2, we could split (ϵ, ϵ) into three boundary pairs: $(\text{country}, \epsilon)$, $(\text{nations}, \epsilon)$, and $(\text{countries}, \epsilon)$. However, it is somewhat inefficient to separately consider the low-scoring child $(\text{countries}, \epsilon)$. Instead, we continue to split off the best child $(\text{country}, \epsilon)$ and leave a note that the zeroth child has been split off, denoted $(\epsilon[1^+], \epsilon)$. The index increases each time a child is split off.

The the boundary pair $(\epsilon[1^+], \epsilon)$ no longer counts $(\text{country}, \epsilon)$ as a child, so its score is lower.

Splitting alternates sides. For example, $(\text{country}, \epsilon)$ splits into $(\text{country}, \text{that})$ and $(\text{country}, \epsilon[1^+])$. If one side has completely revealed words that matter to the language model, then splitting continues with the other side. This procedure ensures that the language model score is completely resolved before considering irrelevant differences, such as coverage of the source sentence.

3.5 Priority Queue

Search proceeds in a best-first fashion controlled by a priority queue. For each source phrase, we convert the compatible hypotheses into a trie. The target phrases were already converted into a trie when the phrase table was loaded. We then push the root (ϵ, ϵ) boundary pair into the priority queue. We do this for all source phrases under consideration, putting their root boundary pairs into the same priority queue. The algorithm then loops by popping the top boundary pair. If the top boundary pair uniquely describes a hypothesis and target phrase, then remaining features are evaluated and the new hypothesis is output to the decoder’s stack. Otherwise, the algorithm splits the boundary pair and pushes both split versions. Iteration continues until k new hypotheses have been found.

3.6 Overall Algorithm

We build hypotheses from left-to-right and manage stacks just like cube pruning. The only difference is how the k elements of these stacks are selected.

When the decoder matches a hypothesis with a compatible source phrase, we immediately evaluate the distortion feature and update future costs, both of which are independent of the target phrase. Our future costs are exactly the same as those used in Moses (Koehn et al., 2007): the highest-scoring way to cover the rest of the source sentence. This includes the language model score within target phrases but ignores the change in language model score that would occur were these phrases to be appended together. The hypotheses compatible with each source phrase are arranged into a trie. Finally, the priority queue algorithm from the preceding section searches for options that the language model likes.

4 Experiments

The primary claim is that our algorithm performs better than cube pruning in terms of the trade-off between time and accuracy. We compare our new decoder implementation with Moses (Koehn et al., 2007) by translating 1677 sentences from Chinese to English. These sentences are a deduplicated subset of the NIST Open MT 2012 test set and were drawn from Chinese online text sources, such as discussion forums. We trained our phrase table using a bitext of 10.8 million sentence pairs, which after tokenization amounts to approximately 290 million words on the English side. The bitext contains data from several sources, including news articles, UN proceedings, Hong Kong government documents, online forum data, and specialized sources such as an idiom translation table. We also trained our language model on the English half of this bitext using unpruned interpolated modified Kneser-Ney smoothing (Kneser and Ney, 1995; Chen and Goodman, 1998).

The system has standard phrase table, length, distortion, and language model features. We plan to implement lexicalized reordering in future work; without this, the test system is 0.53 BLEU (Papineni et al., 2002) point behind a state-of-the-art system. We set the reordering limit to $R = 15$. The phrase table was pre-pruned by applying the same heuristic as Moses: select the top 20 target phrases by score, including the language model.

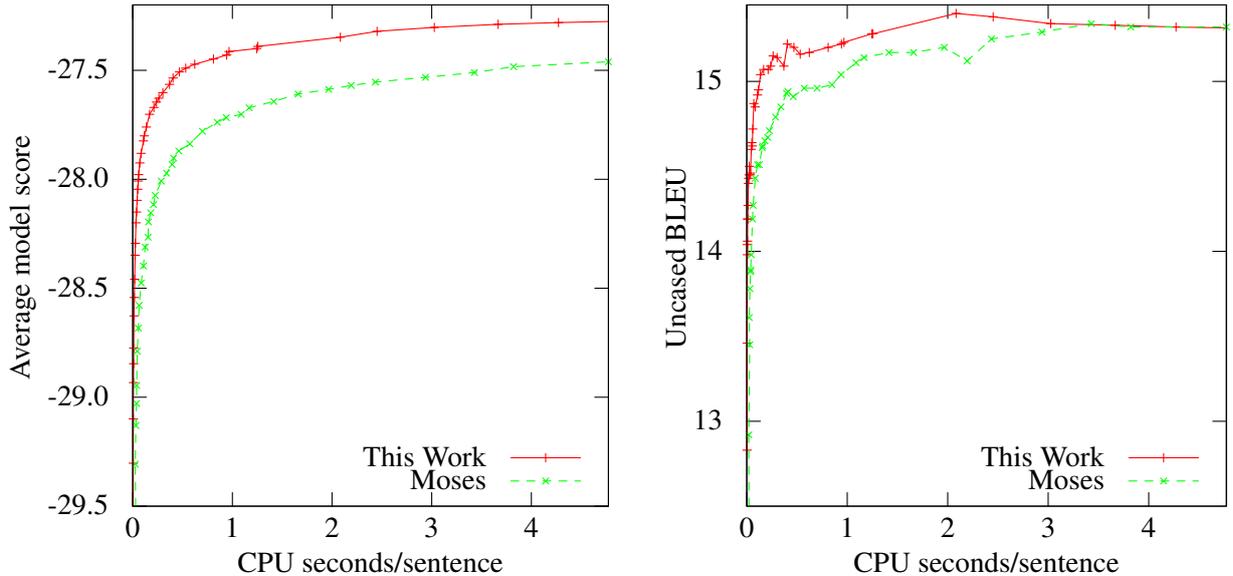


Figure 4: Performance of our decoder and Moses for various stack sizes k .

Moses (Koehn et al., 2007) revision d6df825 was compiled with all optimizations recommended in the documentation. We use the in-memory phrase table for speed. Tests were run on otherwise-idle identical machines with 32 GB RAM; the processes did not come close to running out of memory. The language model was compiled into KenLM probing format (Heafield, 2011) and placed in RAM while text phrase tables were forced into the disk cache before each run. Timing is based on CPU usage (user plus system) minus loading time, as measured by running on empty input; our decoder is also faster at loading. All results are single-threaded. Model score is comparable across decoders and averaged over all 1677 sentences; higher is better. The relationship between model score and uncased BLEU (Papineni et al., 2002) is noisy, so peak BLEU is not attained by the highest search accuracy.

Figure 4 shows the results for pop limits k ranging from 5 to 10000 while Table 1 shows select results. For Moses, we also set the stack size to k to disable a second pruning pass, as is common. Because Moses is slower, we also ran our decoder with higher beam sizes to fill in the graph. Our decoder is more accurate, but mostly faster. We can interpret accuracy improvements as speed improvements by asking how much time is required to attain the same accuracy as the baseline. By this metric, our decoder is 4.0 to 7.7 times as fast as Moses, depending on k .

Stack	Model		CPU		BLEU	
	Moses	This	Moses	This	Moses	This
10	-29.96	-29.70	0.019	0.004	12.92	13.46
100	-28.68	-28.54	0.057	0.016	14.19	14.40
1000	-27.87	-27.80	0.463	0.116	14.91	14.95
10000	-27.46	-27.39	4.773	1.256	15.32	15.28

Table 1: Results for select stack sizes k .

5 Conclusion

We have contributed a new phrase-based search algorithm based on the principle that the language model cares the most about boundary words. This leads to two contributions: hiding irrelevant state from features and an incremental refinement algorithm to find high-scoring combinations. This algorithm is implemented in a new fast phrase-based decoder, which we release as open-source under the LGPL at kheafield.com/code/.

Acknowledgements

This work was supported by the Defense Advanced Research Projects Agency (DARPA) Broad Operational Language Translation (BOLT) program through IBM. This work used Stampede provided by the Texas Advanced Computing Center (TACC) at The University of Texas at Austin under XSEDE allocation TG-CCR140009. XSEDE is supported by NSF grant number OCI-1053575. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA or the US government.

References

- Daniel Cer, Michel Galley, Daniel Jurafsky, and Christopher D. Manning. 2010. Phrasal: A statistical machine translation toolkit for exploring new model features. In *Proceedings of the NAACL HLT 2010 Demonstration Session*, pages 9–12, Los Angeles, California, June. Association for Computational Linguistics.
- Yin-Wen Chang and Michael Collins. 2011. Exact decoding of phrase-based translation models through lagrangian relaxation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, UK, July. Association for Computational Linguistics.
- Stanley Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University, August.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33:201–228, June.
- Kenneth Heafield, Hieu Hoang, Philipp Koehn, Tetsuo Kiso, and Marcello Federico. 2011. Left language model state for syntactic machine translation. In *Proceedings of the International Workshop on Spoken Language Translation*, San Francisco, CA, USA, December.
- Kenneth Heafield, Philipp Koehn, and Alon Lavie. 2013. Grouping language model boundary words to speed k-best extraction from hypergraphs. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, Georgia, USA, June.
- Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, Edinburgh, UK, July. Association for Computational Linguistics.
- Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proceedings of ACL*, Prague, Czech Republic, June.
- Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 181–184.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, Prague, Czech Republic, June.
- Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Machine translation: From real users to research*, pages 115–124. Springer, September.
- Zhifei Li and Sanjeev Khudanpur. 2008. A scalable decoder for parsing-based machine translation with equivalent language model state maintenance. In *Proceedings of the Second ACL Workshop on Syntax and Structure in Statistical Translation (SSST-2)*, pages 10–18, Columbus, Ohio, June.
- Bruce T. Lowerre. 1976. *The Harpy speech recognition system*. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA, USA.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, PA, July.
- Slav Petrov, Aria Haghighi, and Dan Klein. 2008. Coarse-to-fine syntactic machine translation using language projections. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 108–116, Honolulu, HI, USA, October.
- Joern Wuebker, Matthias Huck, Stephan Peitz, Malte Nuhn, Markus Freitag, Jan-Thorsten Peter, Saab Mansour, and Hermann Ney. 2012. Jane 2: Open source phrase-based and hierarchical statistical machine translation. In *Proceedings of COLING 2012: Demonstration Papers*, pages 483–492, Mumbai, India, December.
- Hao Zhang and Daniel Gildea. 2008. Efficient multi-pass decoding for synchronous context free grammars. In *Proceedings of ACL-08: HLT*, pages 209–217, Columbus, Ohio.

Decoder Integration and Expected BLEU Training for Recurrent Neural Network Language Models

Michael Auli

Microsoft Research
Redmond, WA, USA

michael.auli@microsoft.com

Jianfeng Gao

Microsoft Research
Redmond, WA, USA

jfgao@microsoft.com

Abstract

Neural network language models are often trained by optimizing likelihood, but we would prefer to optimize for a task specific metric, such as BLEU in machine translation. We show how a recurrent neural network language model can be optimized towards an expected BLEU loss instead of the usual cross-entropy criterion. Furthermore, we tackle the issue of directly integrating a recurrent network into first-pass decoding under an efficient approximation. Our best results improve a phrase-based statistical machine translation system trained on WMT 2012 French-English data by up to 2.0 BLEU, and the expected BLEU objective improves over a cross-entropy trained model by up to 0.6 BLEU in a single reference setup.

1 Introduction

Neural network-based language and translation models have achieved impressive accuracy improvements on statistical machine translation tasks (Allauzen et al., 2011; Le et al., 2012b; Schwenk et al., 2012; Vaswani et al., 2013; Gao et al., 2014). In this paper we focus on recurrent neural network architectures which have recently advanced the state of the art in language modeling (Mikolov et al., 2010; Mikolov et al., 2011; Sundermeyer et al., 2013) with several subsequent applications in machine translation (Auli et al., 2013; Kalchbrenner and Blunsom, 2013; Hu et al., 2014). Recurrent models have the potential to capture long-span dependencies since their predictions are based on an *unbounded history* of previous words (§2).

In practice, neural network models for machine translation are usually trained by maximizing the likelihood of the training data, either via a cross-entropy objective (Mikolov et al., 2010; Schwenk

et al., 2012) or more recently, noise-contrastive estimation (Vaswani et al., 2013). However, it is widely appreciated that directly optimizing for a task-specific metric often leads to better performance (Goodman, 1996; Och, 2003; Auli and Lopez, 2011). The expected BLEU objective provides an efficient way of achieving this for machine translation (Rosti et al., 2010; Rosti et al., 2011; He and Deng, 2012; Gao and He, 2013; Gao et al., 2014) instead of solely relying on traditional optimizers such as Minimum Error Rate Training (MERT) that only adjust the weighting of entire component models within the log-linear framework of machine translation (§3).

Most previous work on neural networks for machine translation is based on a rescoring setup (Arisoy et al., 2012; Mikolov, 2012; Le et al., 2012a; Auli et al., 2013), thereby side stepping the algorithmic and engineering challenges of direct decoder-integration. One recent exception is Vaswani et al. (2013) who demonstrated that feed-forward network-based language models are more accurate in first-pass decoding than in rescoring. Decoder integration has the advantage for the neural network to directly influence search, unlike rescoring which is restricted to an n-best list or lattice. Decoding with feed-forward architectures is straightforward, since predictions are based on a fixed size input, similar to n-gram language models. However, for recurrent networks we have to deal with the *unbounded history*, which breaks the usual dynamic programming assumptions for efficient search. We show how a simple but effective approximation can side step this issue and we empirically demonstrate its effectiveness (§4).

We test the expected BLEU objective by training a recurrent neural network language model and obtain substantial improvements. We also find that our efficient approximation for decoder integration is very accurate, clearly outperforming a rescoring setup (§5).

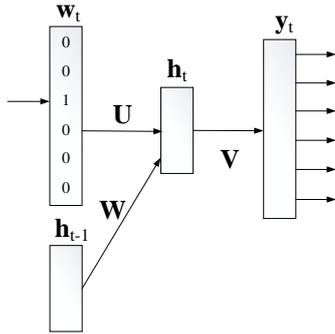


Figure 1: Structure of the recurrent neural network language model.

2 Recurrent Neural Network LMs

Our model has a similar structure to the recurrent neural network language model of Mikolov et al. (2010) which is factored into an input layer, a hidden layer with recurrent connections, and an output layer (Figure 1). The input layer encodes the word at position t as a 1-of- N vector \mathbf{w}_t . The output layer \mathbf{y}_t represents scores over possible next words; both the input and output layers are of size $|V|$, the size of the vocabulary. The hidden layer state \mathbf{h}_t encodes the history of all words observed in the sequence up to time step t . The state of the hidden layer is determined by the input layer and the hidden layer configuration of the previous time step \mathbf{h}_{t-1} . The weights of the connections between the layers are summarized in a number of matrices: \mathbf{U} represents weights from the input layer to the hidden layer, and \mathbf{W} represents connections from the previous hidden layer to the current hidden layer. Matrix \mathbf{V} contains weights between the current hidden layer and the output layer. The activations of the hidden and output layers are computed by:

$$\begin{aligned}\mathbf{h}_t &= \tanh(\mathbf{U}\mathbf{w}_t + \mathbf{W}\mathbf{h}_{t-1}) \\ \mathbf{y}_t &= \tanh(\mathbf{V}\mathbf{h}_t)\end{aligned}$$

Different to previous work (Mikolov et al., 2010), we do not use the softmax activation function to output a probability over the next word, but instead just compute a *single unnormalized score*. This is computationally more efficient than summing over all possible outputs such as required for the cross-entropy error function (Bengio et al., 2003; Mikolov et al., 2010; Schwenk et al., 2012). Training is based on the back propagation through

time algorithm, which unrolls the network and then computes error gradients over multiple time steps (Rumelhart et al., 1986); we use the expected BLEU loss (§3) to obtain the error with respect to the output activations. After training, the output layer represents scores $s(w_{t+1}|w_1 \dots w_t, \mathbf{h}_t)$ for the next word given the previous t input words and the current hidden layer configuration \mathbf{h}_t .

3 Expected BLEU Training

We integrate the recurrent neural network language model as an additional feature into the standard log-linear framework of translation (Och, 2003). Formally, our phrase-based model is parameterized by M parameters Λ where each $\lambda_m \in \Lambda$, $m = 1 \dots M$ is the weight of an associated feature $h_m(f, e)$. Function $h(f, e)$ maps foreign sentences f and English sentences e to the vector $h_1(f, e) \dots (f, e)$, and the model chooses translations according to the following decision rule:

$$\hat{e} = \arg \max_{e \in \mathcal{E}(f)} \Lambda^T h(f, e)$$

We summarize the weights of the recurrent neural network language model as $\theta = \{\mathbf{U}, \mathbf{W}, \mathbf{V}\}$ and add the model as an additional feature to the log-linear translation model using the simplified notation $s_\theta(w_t) = s(w_t|w_1 \dots w_{t-1}, \mathbf{h}_{t-1})$:

$$h_{M+1}(e) = s_\theta(e) = \sum_{t=1}^{|e|} \log s_\theta(w_t) \quad (1)$$

which computes a sentence-level language model score as the sum of individual word scores. The translation model is parameterized by Λ and θ which are learned as follows (Gao et al., 2014):

1. We generate an n-best list for each foreign sentence in the training data with the baseline translation system given Λ where $\lambda_{M+1} = 0$ using the settings described in §5. The n-best lists serve as an approximation to $\mathcal{E}(f)$ used in the next step for expected BLEU training of the recurrent neural network model (§3.1).
2. Next, we fix Λ , set $\lambda_{M+1} = 1$ and optimize θ with respect to the loss function on the training data using stochastic gradient descent (SGD).¹

¹We tuned λ_{M+1} on the development set but found that $\lambda_{M+1} = 1$ resulted in faster training and equal accuracy.

3. We fix θ and re-optimize Λ in the presence of the recurrent neural network model using Minimum Error Rate Training (Och, 2003) on the development set (§5).

3.1 Expected BLEU Objective

Formally, we define our loss function $l(\theta)$ as the negative expected BLEU score, denoted as $\text{xBLEU}(\theta)$ for a given foreign sentence f :

$$\begin{aligned} l(\theta) &= -\text{xBLEU}(\theta) \\ &= \sum_{e \in \mathcal{E}(f)} p_{\Lambda, \theta}(e|f) \text{sBLEU}(e, e^{(i)}) \end{aligned} \quad (2)$$

where $\text{sBLEU}(e, e^{(i)})$ is a smoothed sentence-level BLEU score with respect to the reference translation $e^{(i)}$, and $\mathcal{E}(f)$ is the generation set given by an n-best list.² We use a sentence-level BLEU approximation similar to He and Deng (2012).³ The normalized probability $p_{\Lambda, \theta}(e|f)$ of a particular translation e given f is defined as:

$$p_{\Lambda, \theta}(e|f) = \frac{\exp\{\gamma \Lambda^T h(f, e)\}}{\sum_{e' \in \mathcal{E}(f)} \exp\{\gamma \Lambda^T h(f, e')\}} \quad (3)$$

where $\Lambda^T h(f, e)$ includes the recurrent neural network $h_{M+1}(e)$, and $\gamma \in [0, \text{inf})$ is a scaling factor that flattens the distribution for $\gamma < 1$ and sharpens it for $\gamma > 1$ (Tromble et al., 2008).⁴

Next, we define the gradient of the expected BLEU loss function $l(\theta)$ using the observation that the loss does not explicitly depend on θ :

$$\begin{aligned} \frac{\partial l(\theta)}{\partial \theta} &= \sum_e \sum_{t=1}^{|e|} \frac{\partial l(\theta)}{\partial s_{\theta}(w_t)} \frac{\partial s_{\theta}(w_t)}{\partial \theta} \\ &= \sum_e \sum_{t=1}^{|e|} -\delta_{w_t} \frac{\partial s_{\theta}(w_t)}{\partial \theta} \end{aligned}$$

where δ_{w_t} is the *error term* for English word w_t .⁵ The error term indicates how the loss changes with the translation probability which we derive next.⁶

²Our definitions do not take into account multiple derivations for the same translation because our n-best lists contain only unique entries which we obtain by choosing the highest scoring translation among string identical candidates.

³In early experiments we found that the BLEU+1 approximation used by Liang et al. (2006) and Nakov et. al (2012) worked equally well in our setting.

⁴The γ parameter is *only* used during expected BLEU training but not for subsequent MERT tuning.

⁵A sentence may contain the same word multiple times and we compute the error term for each occurrence separately since the error depends on the individual history.

⁶We omit the gradient of the recurrent neural network score $\frac{\partial s_{\theta}(w_t)}{\partial \theta}$ since it follows the standard form (Mikolov, 2012).

3.2 Derivation of the Error Term δ_{w_t}

We rewrite the loss function (2) using (3) and separate it into two terms $G(\theta)$ and $Z(\theta)$ as follows:

$$\begin{aligned} l(\theta) &= -\text{xBLEU}(\theta) = -\frac{G(\theta)}{Z(\theta)} \\ &= -\frac{\sum_{e \in \mathcal{E}(f)} \exp\{\gamma \Lambda^T h(f, e)\} \text{sBLEU}(e, e^{(i)})}{\sum_{e \in \mathcal{E}(f)} \exp\{\gamma \Lambda^T h(f, e)\}} \end{aligned} \quad (4)$$

Next, we apply the quotient rule of differentiation:

$$\begin{aligned} \delta_{w_t} &= \frac{\partial \text{xBLEU}(\theta)}{\partial s_{\theta}(w_t)} = \frac{\partial (G(\theta)/Z(\theta))}{\partial s_{\theta}(w_t)} \\ &= \frac{1}{Z(\theta)} \left(\frac{\partial G(\theta)}{\partial s_{\theta}(w_t)} - \frac{\partial Z(\theta)}{\partial s_{\theta}(w_t)} \text{xBLEU}(\theta) \right) \end{aligned}$$

Using the observation that θ is only relevant to the recurrent neural network $h_{M+1}(e)$ (1) we have

$$\frac{\partial \gamma \Lambda^T h(f, e)}{\partial s_{\theta}(w_t)} = \gamma \lambda_{M+1} \frac{\partial h_{M+1}(e)}{\partial s_{\theta}(w_t)} = \frac{\gamma \lambda_{M+1}}{s_{\theta}(w_t)}$$

which together with the chain rule, (3) and (4) allows us to rewrite δ_{w_t} as follows:

$$\begin{aligned} \delta_{w_t} &= \frac{1}{Z(\theta)} \sum_{\substack{e \in \mathcal{E}(f), \\ s.t. w_t \in e}} \left(\frac{\partial \exp\{\gamma \Lambda^T h(f, e)\}}{\partial s_{\theta}(w_t)} U(\theta, e) \right) \\ &= \sum_{\substack{e \in \mathcal{E}(f), \\ s.t. w_t \in e}} \left(p_{\Lambda, \theta}(e|f) U(\theta, e) \lambda_{M+1} \frac{\gamma}{s_{\theta}(w_t)} \right) \end{aligned}$$

where $U(\theta, e) = \text{sBLEU}(e, e_i) - \text{xBLEU}(\theta)$.

4 Decoder Integration

Directly integrating our recurrent neural network language model into first-pass decoding enables us to search a much larger space than would be possible in rescoring.

Typically, phrase-based decoders maintain a set of *states* representing partial and complete translation hypothesis that are scored by a set of features. Most features are local, meaning that all required information for them to assign a score is available within the state. One exception is the n-gram language model which requires the preceding $n - 1$ words as well. In order to accommodate this feature, each state usually keeps these words as *context*. Unfortunately, a recurrent neural network makes even weaker independence assumptions so

that it depends on the entire left prefix of a sentence. Furthermore, the weaker independence assumptions also dramatically reduce the effectiveness of dynamic programming by allowing much fewer states to be recombined.⁷

To solve this problem, we follow previous work on lattice rescoring with recurrent networks that maintained the usual n-gram context but kept a beam of hidden layer configurations at each state (Auli et al., 2013). In fact, to make decoding as efficient as possible, we only keep the *single best* scoring hidden layer configuration. This approximation has been effective for lattice rescoring, since the translations represented by each state are in fact very similar: They share both the same source words as well as the same n-gram context which is likely to result in similar recurrent histories that can be safely pruned. As future cost estimate we score each phrase in isolation, resetting the hidden layer at the beginning of a phrase. While simple, we found our estimate to be more accurate than no future cost at all.

5 Experiments

Baseline. We use a phrase-based system similar to Moses (Koehn et al., 2007) based on a set of common features including maximum likelihood estimates $p_{ML}(e|f)$ and $p_{ML}(f|e)$, lexically weighted estimates $p_{LW}(e|f)$ and $p_{LW}(f|e)$, word and phrase-penalties, a hierarchical reordering model (Galley and Manning, 2008), a linear distortion feature, and a modified Kneser-Ney language model trained on the target-side of the parallel data. Log-linear weights are tuned with MERT.

Evaluation. We use training and test data from the WMT 2012 campaign and report results on French-English and German-English. Translation models are estimated on 102M words of parallel data for French-English, and 99M words for German-English; about 6.5M words for each language pair are newswire, the remainder are parliamentary proceedings. We evaluate on six newswire domain test sets from 2008 to 2013 containing between 2034 to 3003 sentences. Log-linear weights are estimated on the 2009 data set comprising 2525 sentences. We evaluate accuracy in terms of BLEU with a single reference.

Rescoring Setup. For rescoring we use ei-

⁷Recombination only retains the highest scoring state if there are multiple identical states, that is, they cover the same source span, the same translation phrase and contexts.

ther lattices or the unique 100-best output of the phrase-based decoder and re-estimate the log-linear weights by running a further iteration of MERT on the n-best list of the development set, augmented by scores corresponding to the neural network models. At test time we rescore n-best lists with the new weights.

Neural Network Training. All neural network models are trained on the news portion of the parallel data, corresponding to 136K sentences, which we found to be most useful in initial experiments. As training data we use unique 100-best lists generated by the baseline system. We use the same data both for training the phrase-based system as well as the language model but find that the resulting bias did not hurt end-to-end accuracy (Yu et al., 2013). The vocabulary consists of words that occur in at least two different sentences, which is 31K words for both language pairs. We tuned the learning rate μ of our mini-batch SGD trainer as well as the probability scaling parameter γ (3) on a held-out set and found simple settings of $\mu = 0.1$ and $\gamma = 1$ to be good choices. To prevent over-fitting, we experimented with L2 regularization, but found no accuracy improvements, probably because SGD regularizes enough. We evaluate performance on a held-out set during training and stop whenever the objective changes less than 0.0003. The hidden layer uses 100 neurons unless otherwise stated.

5.1 Decoder Integration

We compare the effect of direct decoder integration to rescoring with both lattices and n-best lists when the model is trained with a cross-entropy objective (Mikolov et al., 2010). The results (Table 1 and Table 2) show that direct integration improves accuracy across all six test sets on both language pairs. For French-English we improve over n-best rescoring by up to 1.1 BLEU and by up to 0.5 BLEU for German-English. We improve over lattice rescoring by up to 0.4 BLEU on French-English and by up to 0.3 BLEU on German-English. Compared to the baseline, we achieve improvements of up to 2.0 BLEU for French-English and up to 1.3 BLEU for German-English. The average improvement across all test sets is 1.5 BLEU for French-English and 1.0 BLEU for German-English compared to the baseline.

	dev	2008	2010	syscomb2010	2011	2012	2013	AllTest
Baseline	24.11	20.73	24.68	24.59	25.62	24.85	25.54	24.53
RNN n-best rescore	24.83	21.41	25.17	25.06	26.53	25.74	26.31	25.25
RNN lattice rescore	24.91	21.73	25.56	25.43	27.04	26.43	26.75	25.72
RNN decode	25.14	22.03	25.86	25.74	27.32	26.86	27.15	26.06

Table 1: French-English accuracy of decoder integration of a recurrent neural network language model (RNN decode) compared to n-best and lattice rescoring as well as the output of a phrase-based system using an n-gram model (Baseline); Alltest is the corpus-weighted average BLEU across all test sets.

	dev	2008	2010	syscomb2010	2011	2012	2013	AllTest
Baseline	19.35	19.96	20.87	20.66	19.60	19.80	22.48	20.58
RNN n-best rescore	20.17	20.29	21.35	21.27	20.51	20.54	23.03	21.21
RNN lattice rescore	20.24	20.38	21.55	21.43	20.77	20.63	23.23	21.38
RNN decode	20.13	20.51	21.79	21.71	20.91	20.93	23.53	21.61

Table 2: German-English results of direct decoder integration (cf. Table 1).

	dev	2008	2010	syscomb2010	2011	2012	2013	AllTest
Baseline	24.11	20.73	24.68	24.59	25.62	24.85	25.54	24.53
CE RNN	24.80	21.15	25.14	25.06	26.45	25.83	26.69	25.29
+ xBLEU RNN	25.11	21.74	25.52	25.42	27.06	26.42	26.72	25.71

Table 3: French-English accuracy of a decoder integrated cross-entropy recurrent neural network model (CE RNN) and a combination with an expected BLEU trained model (xBLEU RNN). Results are not comparable to Table 1 since a smaller hidden layer was used to keep training times manageable (§5.2).

5.2 Expected BLEU Training

Training with the expected BLEU loss is computationally more expensive than with cross-entropy since each training example is an n-best list instead of a single sentence. This increases the number of words to be processed from 3.5M to 340M. To keep training times manageable, we reduce the hidden layer size to 30 neurons, thereby greatly increasing speed. Despite slower training, the actual scoring at test time of expected BLEU models is about 5 times faster than for cross-entropy models since we do not need to normalize the output layer anymore. The results (Table 3) show improvements of up to 0.6 BLEU when combining a cross-entropy model with an expected BLEU variant. Average gains across all test sets are 0.4 BLEU, demonstrating that the gains from the expected BLEU loss are additive.

6 Conclusion and Future Work

We introduce an empirically effective approximation to integrate a recurrent neural network model into first pass decoding, thereby extending previous work on decoding with feed-forward neu-

ral networks (Vaswani et al., 2013). Our best result improves the output of a phrase-based decoder by up to 2.0 BLEU on French-English translation, outperforming n-best rescoring by up to 1.1 BLEU and lattice rescoring by up to 0.4 BLEU. Directly optimizing a recurrent neural network language model towards an expected BLEU loss proves effective, improving a cross-entropy trained variant by up 0.6 BLEU. Despite higher training complexity, our expected BLEU trained model has five times faster runtime than a cross-entropy model since it does not require normalization.

In future work, we would like to scale up to larger data sets and more complex models through parallelization. We would also like to experiment with more elaborate future cost estimates, such as the average score assigned to all occurrences of a phrase in a large corpus.

7 Acknowledgments

We thank Michel Galley, Arul Menezes, Chris Quirk and Geoffrey Zweig for helpful discussions related to this work as well as the four anonymous reviewers for their comments.

References

- Alexandre Allauzen, H el ene Bonneau-Maynard, Hai-Son Le, Aur elien Max, Guillaume Wisniewski, Fran ois Yvon, Gilles Adda, Josep Maria Crego, Adrien Lardilleux, Thomas Lavergne, and Artem Sokolov. 2011. LIMSI @ WMT11. In *Proc. of WMT*, pages 309–315, Edinburgh, Scotland, July. Association for Computational Linguistics.
- Ebru Arisoy, Tara N. Sainath, Brian Kingsbury, and Bhuvana Ramabhadran. 2012. Deep Neural Network Language Models. In *NAACL-HLT Workshop on the Future of Language Modeling for HLT*, pages 20–28, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Michael Auli and Adam Lopez. 2011. Training a Log-Linear Parser with Loss Functions via Softmax-Margin. In *Proc. of EMNLP*, pages 333–343. Association for Computational Linguistics, July.
- Michael Auli, Michel Galley, Chris Quirk, and Geoffrey Zweig. 2013. Joint Language and Translation Modeling with Recurrent Neural Networks. In *Proc. of EMNLP*, October.
- Yoshua Bengio, R ejean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, 3:1137–1155.
- Michel Galley and Christopher D. Manning. 2008. A Simple and Effective Hierarchical Phrase Reordering Model. In *Proc. of EMNLP*, pages 848–856.
- Jianfeng Gao and Xiaodong He. 2013. Training MRF-Based Phrase Translation Models using Gradient Ascent. In *Proc. of NAACL-HLT*, pages 450–459. Association for Computational Linguistics, June.
- Jianfeng Gao, Xiaodong He, Scott Wen tau Yih, and Li Deng. 2014. Learning Continuous Phrase Representations for Translation Modeling. In *Proc. of ACL*. Association for Computational Linguistics, June.
- Joshua Goodman. 1996. Parsing Algorithms and Metrics. In *Proc. of ACL*, pages 177–183, Santa Cruz, CA, USA, June.
- Xiaodong He and Li Deng. 2012. Maximum Expected BLEU Training of Phrase and Lexicon Translation Models. In *Proc. of ACL*, pages 8–14. Association for Computational Linguistics, July.
- Yuening Hu, Michael Auli, Qin Gao, and Jianfeng Gao. 2014. Minimum Translation Modeling with Recurrent Neural Networks. In *Proc. of EACL*. Association for Computational Linguistics, April.
- Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent Continuous Translation Models. In *Proc. of EMNLP*, pages 1700–1709, Seattle, Washington, USA, October. Association for Computational Linguistics.
- 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 and Poster Sessions*, pages 177–180, Prague, Czech Republic, Jun.
- Hai-Son Le, Alexandre Allauzen, and Fran ois Yvon. 2012a. Continuous Space Translation Models with Neural Networks. In *Proc. of HLT-NAACL*, pages 39–48, Montr eal, Canada. Association for Computational Linguistics.
- Hai-Son Le, Thomas Lavergne, Alexandre Allauzen, Marianna Apidianaki, Li Gong, Aur elien Max, Artem Sokolov, Guillaume Wisniewski, and Fran ois Yvon. 2012b. LIMSI @ WMT12. In *Proc. of WMT*, pages 330–337, Montr eal, Canada, June. Association for Computational Linguistics.
- Percy Liang, Alexandre Bouchard-C ot e, Ben Taskar, and Dan Klein. 2006. An end-to-end discriminative approach to machine translation. In *Proc. of ACL-COLING*, pages 761–768, Jul.
- Tom ař Mikolov, Karafi at Martin, Luk ař Burget, Jan Cernock y, and Sanjeev Khudanpur. 2010. Recurrent Neural Network based Language Model. In *Proc. of INTERSPEECH*, pages 1045–1048.
- Tom ař Mikolov, Anoop Deoras, Daniel Povey, Luk ař Burget, and Jan  ernock y. 2011. Strategies for Training Large Scale Neural Network Language Models. In *Proc. of ASRU*, pages 196–201.
- Tom ař Mikolov. 2012. *Statistical Language Models based on Neural Networks*. Ph.D. thesis, Brno University of Technology.
- Preslav Nakov, Francisco Guzman, and Stephan Vogel. 2012. Optimizing for Sentence-Level BLEU+1 Yields Short Translations. In *Proc. of COLING*. Association for Computational Linguistics.
- Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proc. of ACL*, pages 160–167, Sapporo, Japan, July.
- Antti-Veikko I Rosti, Bing Zhang, Spyros Matsoukas, and Richard Schwartz. 2010. BBN System Description for WMT10 System Combination Task. In *Proc. of WMT*, pages 321–326. Association for Computational Linguistics, July.
- Antti-Veikko I Rosti, Bing Zhang, Spyros Matsoukas, and Richard Schwartz. 2011. Expected BLEU Training for Graphs: BBN System Description for WMT11 System Combination Task. In *Proc. of WMT*, pages 159–165. Association for Computational Linguistics, July.
- David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning Internal Representations by Error Propagation. In *Symposium on Parallel and Distributed Processing*.

- Holger Schwenk, Anthony Rousseau, and Mohammed Attik. 2012. Large, Pruned or Continuous Space Language Models on a GPU for Statistical Machine Translation. In *NAACL-HLT Workshop on the Future of Language Modeling for HLT*, pages 11–19. Association for Computational Linguistics.
- Martin Sundermeyer, Ilya Oparin, Jean-Luc Gauvain, Ben Freiberg, Ralf Schlüter, and Hermann Ney. 2013. Comparison of Feedforward and Recurrent Neural Network Language Models. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 8430–8434, May.
- Roy W. Tromble, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. Lattice Minimum Bayes-Risk Decoding for Statistical Machine Translation. In *Proc. of EMNLP*, pages 620–629. Association for Computational Linguistics, October.
- Ashish Vaswani, Yingdong Zhao, Victoria Fossum, and David Chiang. 2013. Decoding with Large-scale Neural Language Models improves Translation. In *Proc. of EMNLP*. Association for Computational Linguistics, October.
- Heng Yu, Liang Huang, Haitao Mi, and Kai Zhao. 2013. Max-Violation Perceptron and Forced Decoding for Scalable MT Training. In *Proc. of EMNLP*, pages 1112–1123. Association for Computational Linguistics, October.

On the Elements of an Accurate Tree-to-String Machine Translation System

Graham Neubig, Kevin Duh

Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama-cho, Ikoma-shi, Nara, Japan
{neubig, kevinduh}@is.naist.jp

Abstract

While tree-to-string (T2S) translation theoretically holds promise for efficient, accurate translation, in previous reports T2S systems have often proven inferior to other machine translation (MT) methods such as phrase-based or hierarchical phrase-based MT. In this paper, we attempt to clarify the reason for this performance gap by investigating a number of peripheral elements that affect the accuracy of T2S systems, including parsing, alignment, and search. Based on detailed experiments on the English-Japanese and Japanese-English pairs, we show how a basic T2S system that performs on par with phrase-based systems can be improved by 2.6-4.6 BLEU, greatly exceeding existing state-of-the-art methods. These results indicate that T2S systems indeed hold much promise, but the above-mentioned elements must be taken seriously in construction of these systems.

1 Introduction

In recent years, syntactic parsing is being viewed as an ever-more important element of statistical machine translation (SMT) systems, particularly for translation between languages with large differences in word order. There are many ways of incorporating syntax into MT systems, including the use of string-to-tree translation (S2T) to ensure the syntactic well-formedness of the output (Galley et al., 2006; Shen et al., 2008), tree-to-string (T2S) using source-side parsing as a hint during the translation process (Liu et al., 2006), or pre- or post-ordering to help compensate for reordering problems experienced by non-syntactic methods such as phrase-based MT (PBMT) (Collins et al., 2005; Sudoh et al., 2011). Among these, T2S

translation has a number of attractive theoretical properties, such as joint consideration of global re-ordering and lexical choice while maintaining relatively fast decoding times.

However, building an accurate T2S system is not trivial. On one hand, there have been multiple reports (mainly from groups with a long history of building T2S systems) stating that systems using source-side syntax greatly out-perform phrase-based systems (Mi et al., 2008; Liu et al., 2011; Zhang et al., 2011; Tamura et al., 2013). On the other hand, there have been also been multiple reports noting the exact opposite result that source-side syntax systems perform worse than Hiero, S2T, PBMT, or PBMT with pre-ordering (Ambati and Lavie, 2008; Xie et al., 2011; Kaljahi et al., 2012). In this paper, we argue that this is due to the fact that T2S systems have the potential to achieve high accuracy, but are also less robust, with a number of peripheral elements having a large effect on translation accuracy.

Our motivation in writing this paper is to provide a first step in examining and codifying the more important elements that make it possible to construct a highly accurate T2S MT system. To do so, we perform an empirical study of the effect of parsing accuracy, packed forest input, alignment accuracy, and search. The reason why we choose these elements is that past work that has reported low accuracy for T2S systems has often neglected to consider one or all of these elements.

As a result of our tests on English-Japanese (en-ja) and Japanese-English (ja-en) machine translation, we find that a T2S system not considering these elements performs only slightly better than a standard PBMT system. However, after accounting for all these elements we see large increases of accuracy, with the final system greatly exceeding not only standard PBMT, but also state-of-the-art methods based on syntactic pre- or post-ordering.

2 Experimental Setup

2.1 Systems Compared

In our experiments, we use a translation model based on T2S tree transducers (Graehl and Knight, 2004), constructed using the Travatar toolkit (Neubig, 2013). Rules are extracted using the GHKM algorithm (Galley et al., 2006), and rules with up to 5 composed minimal rules, up to 2 non-terminals, and up to 10 terminals are used.

We also prepare 3 baselines not based on T2S to provide a comparison with other systems in the literature. The first two baselines are standard systems using PBMT or Hiero trained using Moses (Koehn et al., 2007). We use default settings, except for setting the reordering limit or maximum chart span to the best-performing value of 24. As our last baselines, we use two methods based on syntactic pre- or post-ordering, which are state-of-the-art methods for the language pairs. Specifically, for en-ja translation we use the head finalization pre-ordering method of (Isozaki et al., 2010b), and for ja-en translation, we use the syntactic post-ordering method of (Goto et al., 2012). For all systems, T2S or otherwise, the language model is a Kneser-Ney 5-gram, and tuning is performed to maximize BLEU score using minimum error rate training (Och, 2003).

2.2 Data and Evaluation

We perform all of our experiments on en-ja and ja-en translation over data from the NTCIR PatentMT task (Goto et al., 2011), the most standard benchmark task for these language pairs. We use the training data from NTCIR 7/8, a total of approximately 3.0M sentences, and perform tuning on the NTCIR 7 dry run, testing on the NTCIR 7 formal run data. As evaluation measures, we use the standard BLEU (Papineni et al., 2002) as well as RIBES (Isozaki et al., 2010a), a reordering-based metric that has been shown to have high correlation with human evaluations on the NTCIR data. We measure significance of results using bootstrap resampling at $p < 0.05$ (Koehn, 2004). In tables, bold numbers indicate the best system and all systems that were not significantly different from the best system.

2.3 Motivational Experiment

Before going into a detailed analysis, we first present results that stress the importance of the elements described in the introduction. To do so,

System	en-ja		ja-en	
	BLEU	RIBES	BLEU	RIBES
PBMT	35.84	72.89	30.49	69.80
Hiero	34.45	72.94	29.41	69.51
Pre/Post	36.69	77.05	29.42	73.85
T2S-all	36.23	76.60	31.15	72.87
T2S+all	40.84	80.15	33.70	75.94

Table 1: Overall results for five systems.

we compare the 3 non-T2S baselines with two T2S systems that vary the settings of the parser, alignment, and search, as described in the following Sections 3, 4, and 5. The first system “T2S-all” is a system that uses the worst settings¹ for each of these elements, while the second system “T2S+all” uses the best settings.² The results for the systems are shown in Table 1.

The most striking result is that T2S+all significantly exceeds all of the baselines, even including the pre/post-ordering baselines, which provide state-of-the-art results on this task. The gains are particularly striking on en-ja, with a gain of over 4 BLEU points over the closest system, but still significant on the ja-en task, where the use of source-side syntax has proven less effective in previous work (Sudoh et al., 2011). The next thing to notice is that if we had instead used T2S-all, our conclusion would have been much different. This system is able to achieve respectable accuracy compared to PBMT or Hiero, but does not exceed the more competitive pre/post-ordering systems.³ With this result in hand, we will investigate the contribution of each of these elements in detail in the following sections. In the remainder of the paper settings follow T2S+all except when otherwise noted.

3 Parsing

3.1 Parsing Overview

As T2S translation uses parse trees both in training and testing of the system, an accurate syntactic parser is required. In order to test the extent that parsing accuracy affects translation, we use two

¹Stanford/Eda, GIZA++, pop-limit 5000 cube pruning.

²Egret forests, Nile, pop-limit 5000 hypergraph search.

³We have also observed similar trends on other genres and language pairs. For example, in a Japanese-Chinese/English medical conversation task (Neubig et al., 2013), forests, alignment, and search resulted in BLEU increases of en-ja 24.55→30.81, ja-en 19.28→22.46, zh-ja 15.22→20.67, ja-zh 30.88→33.89.

different syntactic parsers and examine the translation accuracy realized by each parser.

For English, the two most widely referenced parsers are the Stanford Parser and Berkeley Parser. In this work, we compare the Stanford Parser’s CFG model, with the Berkeley Parser’s latent variable model. In previous reports, it has been noted (Kummerfeld et al., 2012) that the latent variable model of the Berkeley parser tends to have the higher accuracy of the two, so if the accuracy of a system using this model is higher then it is likely that parsing accuracy is important for T2S translation. Instead of the Berkeley Parser itself, we use a clone Egret,⁴ which achieves nearly identical accuracy, and is able to output packed forests for use in MT, as mentioned below. Trees are right-binarized, with the exception of phrase-final punctuation, which is split off before any other element in the phrase.

For Japanese, our first method uses the MST-based pointwise dependency parser of Flannery et al. (2011), as implemented in the Eda toolkit.⁵ In order to convert dependencies into phrase-structure trees typically used in T2S translation, we use the head rules implemented in the Travatar toolkit. In addition, we also train a latent variable CFG using the Berkeley Parser and use Egret for parsing. Both models are trained on the Japanese Word Dependency Treebank (Mori et al., 2014).

In addition, Mi et al. (2008) have proposed a method for forest-to-string (F2S) translation using packed forests to encode many possible sentence interpretations. By doing so, it is possible to resolve some of the ambiguity in syntactic interpretation at translation time, potentially increasing translation accuracy. However, the great majority of recent works on T2S translation do not consider multiple syntactic parses (e.g. Liu et al. (2011), Zhang et al. (2011)), and thus it is important to confirm the potential gains that could be acquired by taking ambiguity into account.

3.2 Effect of Parsing and Forest Input

In Table 2 we show the results for Stanford/Eda with 1-best tree input vs. Egret with trees or forests as input. Forests are those containing all edges in the 100-best parses.

First looking at the difference between the two parsers, we can see that the T2S system using

⁴<http://code.google.com/p/egret-parser>

⁵<http://plata.ar.media.kyoto-u.ac.jp/tool/EDA>

System	en-ja		ja-en	
	BLEU	RIBES	BLEU	RIBES
Stan/Eda	38.95	78.47	32.56	73.03
Egret-T	39.26	79.26	32.97	74.94
Egret-F	40.84	80.15	33.70	75.94

Table 2: Results for Stanford/Eda, Egret with tree input, and Egret with forest input.

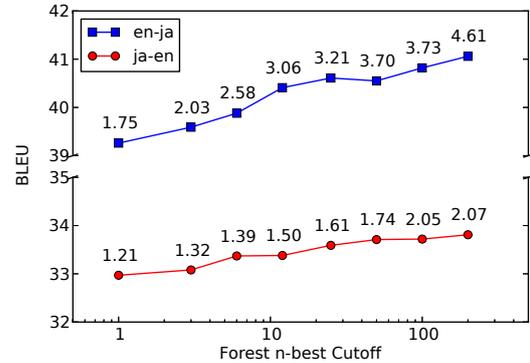


Figure 1: BLEU scores using various levels of forest pruning. Numbers in the graph indicate decoding time in seconds/sentence.

Egret achieves greater accuracy than that using the other two parsers. This improvement is particularly obvious in RIBES, indicating that an increase in parsing accuracy has a larger effect on global reordering than on lexical choice. When going from T2S to F2S translation using Egret, we see another large gain in accuracy, although this time with the gain in BLEU being more prominent. We believe this is related to the observation of Zhang and Chiang (2012) that F2S translation is not necessarily helping fixing parsing errors, but instead giving the translation system the freedom to ignore the parse somewhat, allowing for less syntactically motivated but more fluent translations.

As passing some degree of syntactic ambiguity on to the decoder through F2S translation has proven useful, a next natural question is how much of this ambiguity we need to preserve in our forest. The pruning criterion that we use for the forest is based on including all edges that appear in one or more of the n -best parses, so we perform translation setting n to 1 (trees), 3, 6, 12, 25, 50, 100, and 200. Figure 1 shows results for these settings with regards to translation accuracy and speed. Overall, we can see that every time we double the size of the forest we get an approximately linear in-

crease in BLEU at the cost of an increase in decoding time. Interestingly, the increases in BLEU did not show any sign of saturating even when setting the n -best cutoff to 200, although larger cutoffs resulted in exceedingly large translation forests that required large amounts of memory.

4 Alignment

4.1 Alignment Overview

The second element that we investigate is alignment accuracy. It has been noted in many previous works that significant gains in alignment accuracy do not make a significant difference in translation results (Ayan and Dorr, 2006; Ganchev et al., 2008). However, none of these works have explicitly investigated the effect on T2S translation, so it is not clear whether these results carry over to our current situation.

As our baseline aligner, we use the GIZA++ implementation of the IBM models (Och and Ney, 2003) with the default options. To test the effect of improved alignment accuracy, we use the discriminative alignment method of Riesa and Marcu (2010) as implemented in the Nile toolkit.⁶ This method has the ability to use source- and target-side syntactic information, and has been shown to improve the accuracy of S2T translation.

We trained Nile and tested both methods on the Japanese-English alignments provided with the Kyoto Free Translation Task (Neubig, 2011) (430k parallel sentences, 1074 manually aligned training sentences, and 120 manually aligned test sentences).⁷ As creating manual alignment data is costly, we also created two training sets that consisted of 1/4 and 1/16 of the total data to test if we can achieve an effect with smaller amounts of manually annotated data. The details of data size and alignment accuracy are shown in Table 3.

4.2 Effect of Alignment on Translation

In Table 4, we show results when we vary the aligner between GIZA++ and Nile. For reference, we also demonstrate results when using the same alignments for PBMT and Hiero.

From this, we can see that while for PBMT and Hiero systems the results are mixed, as has been noted in previous work (Fraser and Marcu, 2007),

⁶<http://code.google.com/p/nile>

⁷This data is from Wikipedia articles about Kyoto City, and is an entirely different genre than our MT test data. It is likely that creating aligned data that matches the MT genre would provide larger gains in MT accuracy.

Name	Sent.	Prec.	Rec.	F-meas
GIZA++	0	60.46	55.48	57.86
Nile/16	68	70.21	60.81	65.17
Nile/4	269	72.85	62.70	67.40
Nile	1074	72.73	63.97	68.07

Table 3: Alignment accuracy (%) by method and number of manually annotated training sentences.

System	en-ja		ja-en	
	BLEU	RIBES	BLEU	RIBES
PBMT-G	35.84	72.89	30.49	69.80
PBMT-N	36.05	71.84	30.77	69.75
Hiero-G	34.45	72.94	29.41	69.51
Hiero-N	33.90	72.63	28.90	69.83
T2S-G	39.57	78.94	32.62	75.19
T2S-N/16	40.79	80.05	32.82	74.89
T2S-N/4	40.97	80.32	33.35	75.46
T2S-N	40.84	80.15	33.70	75.94

Table 4: Results varying the aligner (GIZA++ vs. Nile), including results for Nile when using 1/4 or 1/16 of the annotated training data.

		good		bad
VP → VBZ ₁ NP ₂		X ₂ X ₁	X ₂ wo X ₁	X ₁ X ₂
P(ef)	GIZA++	0.16	0.01	0.78
	Nile	0.30	0.47	0.05

Figure 2: Probabilities for SVO→SOV rules.

improving the alignment accuracy gives significant gains for T2S translation. The reason for this difference is two-fold. The first is that in rule extraction in syntax-based translation (Galley et al., 2006), a single mistaken alignment crossing phrase boundaries results not only in a bad rule being extracted, but also prevents the extraction of a number of good rules. This is reflected in the size of the rule table; the en-ja system built using Nile contains 92.8M rules, while the GIZA++ system contains only 83.3M rules, a 11.2% drop.

The second reason why alignment is important is that while one of the merits of T2S models is their ability to perform global re-ordering, it is difficult to learn good reorderings from bad alignments. We show an example of this in Figure 2. When translating SVO English to SOV Japanese, we expect rules containing a verb and a following noun phrase (VO) to have a high probability of being reversed (to OV), possibly with the addition of

the Japanese direct object particle “wo.” From the figure, we can see that the probabilities learned by Nile match this intuition, while the probabilities learned by GIZA heavily favor no reordering.

Finally, looking at the amount of data needed to train the model, we can see that a relatively small amount of manually annotated data proves sufficient for large gains in alignment accuracy, with even 68 sentences showing a 7.31 point gain in F-measure over GIZA++. This is because Nile’s feature set uses generalizable POS/syntactic information and also because mis-alignments of common function words (e.g. a/the) will be covered even by small sets of training data. Looking at the MT results, we can see that even the smaller data sets allow for gains in accuracy, although the gains are more prominent for en-ja.

5 Search

5.1 Search Overview

Finally, we examine the effect that the choice of search algorithm has on the accuracy of translation. The most standard search algorithm for T2S translation is bottom-up beam search using cube pruning (CP, Chiang (2007)). However, there are a number of other search algorithms that have been proposed for tree-based translation in general (Huang and Chiang, 2007) or T2S systems in particular (Huang and Mi, 2010; Feng et al., 2012). In this work, we compare CP and the hypergraph search (HS) method of Heafield et al. (2013), which is also a bottom-up pruning algorithm but performs more efficient search by grouping together similar language model states.

5.2 Effect of Search

Figure 3 shows BLEU and decoding speed results using HS or CP on T2S and F2S translation, using a variety of pop limits. From this, we can see that HS out-performs CP for both F2S and T2S, especially with smaller pop limits. Comparing the graphs for F2S and T2S translation, it is notable that the shapes of the graphs for the two methods are strikingly similar. This result is somewhat surprising, as the overall search space of F2S is larger and it would be natural for the characteristics of the search algorithm to vary between these two settings. Finally, comparing ja-en and en-ja, search is simpler for the former, a result of the fact that the Japanese sentences contain more words, and thus more LM evaluations per sentence.

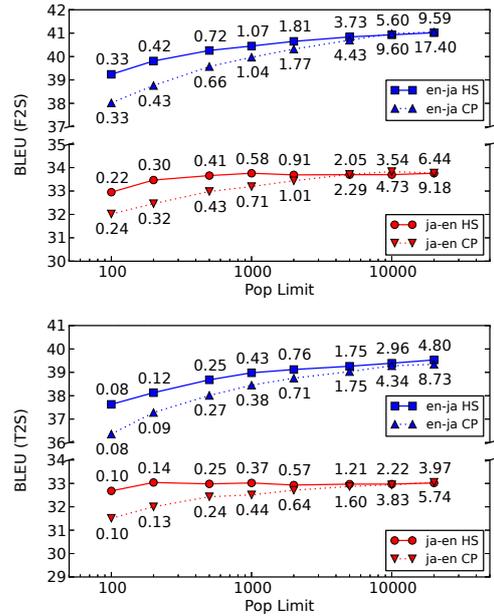


Figure 3: Hypergraph search (HS) and cube pruning (CP) results for F2S and T2S. Numbers above and below the lines indicate time in seconds/sentence for HS and CP respectively.

6 Conclusion

In this paper, we discussed the importance of three peripheral elements that contribute greatly to the accuracy of T2S machine translation: parsing, alignment, and search. Put together, a T2S system that uses the more effective settings for these three elements greatly outperforms a system that uses more standard settings, as well as the current state-of-the-art on English-Japanese and Japanese-English translation tasks.

Based on these results we draw three conclusions. The first is that given the very competitive results presented here, T2S systems do seem to have the potential to achieve high accuracy, even when compared to strong baselines incorporating syntactic reordering into a phrase-based system. The second is that when going forward with research on T2S translation, one should first be sure to account for these three elements to ensure a sturdy foundation for any further improvements. Finally, considering the fact that parsing and alignment for each of these languages is far from perfect, further research investment in these fields may very well have the potential to provide additional gains in accuracy in the T2S framework.

Acknowledgments: This work was supported by JSPS KAKENHI Grant Number 25730136.

References

- Vamshi Ambati and Alon Lavie. 2008. Improving syntax driven translation models by re-structuring divergent and non-isomorphic parse tree structures. In *Proc. AMTA*, pages 235–244.
- Necip Ayan and Bonnie Dorr. 2006. Going beyond AER: an extensive analysis of word alignments and their impact on MT. In *Proc. ACL*.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.
- Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proc. ACL*, pages 531–540.
- Yang Feng, Yang Liu, Qun Liu, and Trevor Cohn. 2012. Left-to-right tree-to-string decoding with prediction. In *Proc. EMNLP*, pages 1191–1200.
- Daniel Flannery, Yusuke Miyao, Graham Neubig, and Shinsuke Mori. 2011. Training dependency parsers from partially annotated corpora. In *Proc. IJCNLP*, pages 776–784.
- Alexander Fraser and Daniel Marcu. 2007. Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, 33(3):293–303.
- Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proc. ACL*, pages 961–968.
- Kuzman Ganchev, João V. Graça, and Ben Taskar. 2008. Better alignments = better translations? In *Proc. ACL*.
- Isao Goto, Bin Lu, Ka Po Chow, Eiichiro Sumita, and Benjamin K. Tsou. 2011. Overview of the patent machine translation task at the NTCIR-9 workshop. In *Proceedings of NTCIR*, volume 9, pages 559–578.
- Isao Goto, Masao Utiyama, and Eiichiro Sumita. 2012. Post-ordering by parsing for Japanese-English statistical machine translation. In *Proc. ACL*, pages 311–316.
- Jonathan Graehl and Kevin Knight. 2004. Training tree transducers. In *Proc. HLT*, pages 105–112.
- Kenneth Heafield, Philipp Koehn, and Alon Lavie. 2013. Grouping language model boundary words to speed k-best extraction from hypergraphs. In *Proc. NAACL*, pages 958–968.
- Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proc. ACL*, pages 144–151.
- Liang Huang and Haitao Mi. 2010. Efficient incremental decoding for tree-to-string translation. In *Proc. EMNLP*, pages 273–283.
- Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010a. Automatic evaluation of translation quality for distant language pairs. In *Proc. EMNLP*, pages 944–952.
- Hideki Isozaki, Katsuhito Sudoh, Hajime Tsukada, and Kevin Duh. 2010b. Head finalization: A simple reordering rule for SOV languages. In *Proc. WMT and MetricsMATR*.
- Rasoul Samad Zadeh Kaljahi, Raphael Rubino, Johann Roturier, and Jennifer Foster. 2012. A detailed analysis of phrase-based and syntax-based machine translation: The search for systematic differences. In *Proc. AMTA*.
- 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. ACL*, pages 177–180.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. EMNLP*.
- Jonathan K Kummerfeld, David Hall, James R Curran, and Dan Klein. 2012. Parser showdown at the wall street corral: an empirical investigation of error types in parser output. In *Proc. EMNLP*, pages 1048–1059.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. ACL*.
- Yang Liu, Qun Liu, and Yajuan Lü. 2011. Adjoining tree-to-string translation. In *Proc. ACL*, pages 1278–1287.
- Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proc. ACL*, pages 192–199.
- Shinsuke Mori, Hideki Ogura, and Tetsuro Sasada. 2014. A Japanese word dependency corpus. In *Proc. LREC*.
- Graham Neubig, Sakriani Sakti, Tomoki Toda, Satoshi Nakamura, Yuji Matsumoto, Ryosuke Isotani, and Yukichi Ikeda. 2013. Towards high-reliability speech translation in the medical domain. In *Proc. MedNLP*, pages 22–29.
- Graham Neubig. 2011. The Kyoto free translation task. <http://www.phontron.com/kfft>.
- Graham Neubig. 2013. Travatar: A forest-to-string machine translation engine based on tree transducers. In *Proc. ACL Demo Track*, pages 91–96.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. ACL*, pages 160–167.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. ACL*, pages 311–318.
- Jason Riesa and Daniel Marcu. 2010. Hierarchical search for word alignment. In *Proc. ACL*, pages 157–166.
- Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proc. ACL*, pages 577–585.
- Katsuhito Sudoh, Xianchao Wu, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. 2011. Post-ordering in statistical machine translation. In *Proc. MT Summit*.
- Akihiro Tamura, Taro Watanabe, Eiichiro Sumita, Hiroya Takamura, and Manabu Okumura. 2013. Part-of-speech induction in dependency trees for statistical machine translation. In *Proc. ACL*, pages 841–851.
- Jun Xie, Haitao Mi, and Qun Liu. 2011. A novel dependency-to-string model for statistical machine translation. In *Proc. EMNLP*, pages 216–226.
- Hui Zhang and David Chiang. 2012. An exploration of forest-to-string translation: Does translation help or hurt parsing? In *Proc. ACL*, pages 317–321.
- Hao Zhang, Licheng Fang, Peng Xu, and Xiaoyun Wu. 2011. Binarized forest to string translation. In *Proc. ACL*, pages 835–845.

Simple extensions for a reparameterised IBM Model 2

Douwe Gelling

Department of Computer Science
The University of Sheffield
d.gelling@shef.ac.uk

Trevor Cohn

Computing and Information Systems
The University of Melbourne
t.cohn@unimelb.edu.au

Abstract

A modification of a reparameterisation of IBM Model 2 is presented, which makes the model more flexible, and able to model a preference for aligning to words to either the right or left, and take into account POS tags on the target side of the corpus. We show that this extension has a very small impact on training times, while obtaining better alignments in terms of BLEU scores.

1 Introduction

Word alignment is at the basis of most statistical machine translation. The models that are generally used are often slow to train, and have a large number of parameters. Dyer et al. (2013) present a simple reparameterization of IBM Model 2 that is very fast to train, and achieves results similar to IBM Model 4.

While this model is very effective, it also has a very low number of parameters, and as such doesn't have a large amount of expressive power. For one thing, it forces the model to consider alignments on both sides of the diagonal equally likely. However, it isn't clear that this is the case, as for some languages an alignment to earlier or later in the sentence (above or below the diagonal) could be common, due to word order differences. For example, when aligning to Dutch, it may be common for one verb to be aligned near the end of the sentence that would be at the beginning in English. This would mean most of the other words in the sentence would also align slightly away from the diagonal in one direction. Figure 1 shows an example sentence in which this happens. Here, a circle denotes an alignment, and darker squares are more likely under the alignment model. In this case the modified Model 2 would simply make both directions equally likely, where we would really like for only one direction to be more likely.

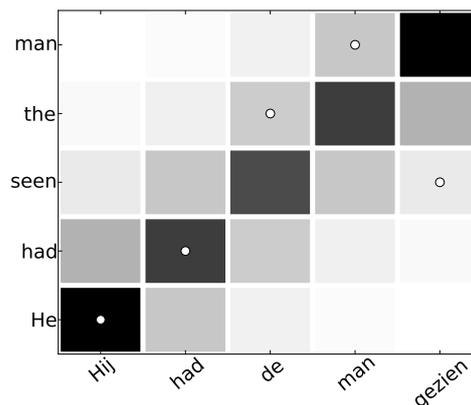


Figure 1: Visualization of aligned sentence pair in Dutch and English, darker shaded squares have a higher alignment probability under the model, a circle indicates a correct alignment. The English sentence runs from bottom to top, the Dutch sentence left to Right.

In some cases it could be that the prior probability for a word alignment should be off the diagonal.

Furthermore, it is common in word alignment to take word classes into account. This is commonly implemented for the HMM alignment model as well as Models 4 and 5. Och and Ney (2003) show that for larger corpora, using word classes leads to lower Alignment Error Rate (AER). This is not implemented for Model 2, as it already has an alignment model that is dependent on both source and target length, and the position in both sentences, and adding a dependency to word classes would make the the Model even more prone to overfitting than it already is. However, using the reparameterization in (Dyer et al., 2013) would leave the model simple enough even with a relatively large amount of word classes.

Figure 2 shows an example of how the model extensions could benefit word alignment. In the example, all the Dutch words have a different

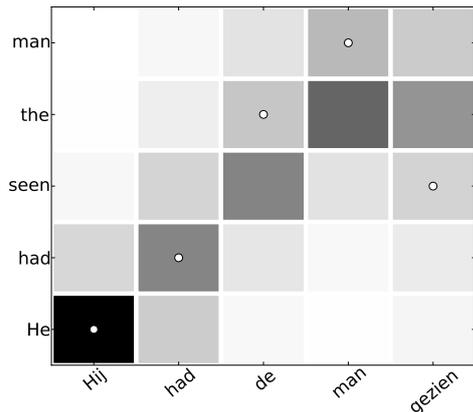


Figure 2: Visualization of aligned sentence pair in Dutch and English, darker shaded squares have a higher alignment probability under the model, a circle indicates a correct alignment. The English sentence runs from bottom to top, the Dutch sentence left to Right.

word class, and so can have different gradients for alignment probability over the english words. If the model has learned that prepositions and nouns are more likely to align to words later in the sentence, it could have a lower lambda for both word classes, resulting in a less steep slope. If we also split lambda into two variables, we can get alignment probabilities as shown above for the Dutch word 'de', where aligning to one side of the diagonal is made more likely for some word classes. Finally, instead of just having one side of the diagonal less steep than the other, it may be useful to instead move the peak of the alignment probability function off the diagonal, while keeping it equally likely. In Figure 2, this is done for the past participle 'gezien'.

We will present a simple model for adding the above extensions to achieve the above (splitting the parameter, adding an offset and conditioning the parameters on the POS tag of the target word) in section 2, results on a set of experiments in section 3 and present our conclusions in section 4.

2 Methods

We make use of a modified version of Model 2, from Dyer et al. (2013), which has an alignment model that is parameterised in its original form solely on the variable λ . Specifically, the probability of a sentence \mathbf{e} given a sentence \mathbf{f} is given as:

$$\prod_{i=1}^m \sum_{j=0}^n \delta(a_i|i, m, n) \cdot \theta(e_i|f_{a_i})$$

here, m is the length of the target sentence \mathbf{e} , n the same for source sentence \mathbf{f} , δ is the alignment model and θ is the translation model. In this paper we are mainly concerned with the alignment model δ . In the original formulation (with a minor tweak to ensure symmetry through the center), this function is defined as:

$$\delta(a_i = j|i, m, n) = \begin{cases} p_0 & j = 0 \\ (1 - p_0) \cdot \frac{e^{h(i,j,m,n)}}{Z(i,m,n)} & 0 < j \leq n \\ 0 & \text{otherwise} \end{cases}$$

where, $h(\cdot)$ is defined as

$$h(i, j, m, n) = -\lambda \left| \frac{i}{m+1} - \frac{j}{n+1} \right|$$

and $Z_\lambda(i, m, n)$ is $\sum_{j'=1}^n e^{\lambda h(i,j',m,n)}$, i.e. a normalising function. Like the original Model 2 (Brown et al., 1993), this model is trained using Expectation-Maximisation. However, it is not possible to directly update the λ parameter during training, as it cannot be computed analytically. Instead, a gradient-based approach is used during the M-step.

Two different optimisations are employed, the first of which is used for calculating Z_λ . This function forms a geometric series away from the diagonal (for each target word), which can be computed efficiently for each of the directions from the diagonal. The second is used during the M-step when computing the derivative, and is very similar, but instead of using a geometric series, an arithmetico-geometric series is used.

In order to allow the model to have a different parameter above and below the diagonal, the only change needed is to redefine $h(\cdot)$ to use a different parameter for λ above and below the diagonal. We denote these parameters as λ and γ for below and above the diagonal respectively. Further, the offset is denoted as ω .

we change the definition of $h(\cdot)$ to the following instead:

$$h(i, j, m, n) = \begin{cases} -\lambda \left| \frac{i}{m+1} - \frac{j}{n+1} + \omega \right| & j \leq j_{\downarrow} \\ -\gamma \left| \frac{i}{m+1} - \frac{j}{n+1} + \omega \right| & \text{otherwise} \end{cases}$$

j_{\downarrow} is the point closest to or on the diagonal here, calculated as:

$$\max(\min(\lfloor \frac{i \cdot (n+1)}{m+1} + \omega \cdot (n+1) \rfloor, n), 0)$$

Here, ω can range from -1 to 1 , and thus the calculation for the diagonal j_{\downarrow} is clamped to be in a valid range for alignments.

As the partition function ($Z(\cdot)$) used in (Dyer et al., 2013) consists of 2 calculations for each target position i , one for above and one for below the diagonal, we can simply substitute γ for the geometric series calculations in order to use different parameters for each:

$$s_{\downarrow}(e^{\lambda h(i, j_{\downarrow}, m, n)}, r) + s_{n-\uparrow}(e^{\gamma h(i, j_{\uparrow}, m, n)}, r)$$

where j_{\uparrow} is $j_{\downarrow} + 1$.

2.1 Optimizing the Parameters

As in the original formulation, we need to use gradient-based optimisation in order to find good values for λ , γ and ω . Unfortunately, optimizing ω would require taking the derivative of $h(\cdot)$, and thus the derivative of the absolute value. This is unfortunately undefined when the argument is 0, however we work around this by choosing a sub-gradient of 0 at that point. This means the steps we take do not always improve the objective function, but in practice the method works well.

The first derivative of \mathcal{L} with respect to λ at a single target word becomes:

$$\nabla_{\lambda} \mathcal{L} = \sum_{k=1}^{j_{\downarrow}} p(a_i = k | e_i, \mathbf{f}, m, n) h(i, k, m, n) - \sum_{l=1}^{j_{\downarrow}} \delta(l | i, m, n) h(i, l, m, n)$$

And similar for finding the first derivative with respect to γ , but summing from j_{\uparrow} to n instead. The first derivative with respect to ω then, is:

$$\nabla_{\omega} \mathcal{L} = \sum_{k=1}^n p(a_i = k | e_i, \mathbf{f}, m, n) h'(i, k, m, n) - \sum_{l=1}^{j_{\downarrow}} \delta(l | i, m, n) h'(i, l, m, n)$$

Where $h'(\cdot)$ is the first derivative of $h(\cdot)$ with respect to ω . For obtaining this derivative, the arithmetico-geometric series (Fernandez et al., 2006) was originally used as an optimization, and for the gradient with respect to *omega* a geometric series should suffice, as an optimization, as there is no conditioning on the source words. This is not done in the current work however, so timing results will not be directly comparable to those found in (Dyer et al., 2013).

Conditioning on the POS of the target words then becomes as simple as using a different λ , γ , and ω for each POS tag in the input, and calculating a separate derivative for each of them, using only the derivatives at those target words that use the POS tag. A minor detail is to keep a count of alignment positions used for finding the derivative for each different parameter, and normalizing the resulting derivatives with those counts, so the step size can be kept constant across POS tags.

3 Empirical results

The above described model is evaluated with experiments on a set of 3 language pairs, on which AER scores and BLEU scores are computed. We use similar corpora as used in (Dyer et al., 2013): a French-English corpus made up of Europarl version 7 and news-commentary corpora, the Arabic-English parallel data consisting of the non-UN portions of the NIST training corpora, and the FBIS Chinese-English corpora.

The models that are compared are the original reparameterization of Model 2, a version where λ is split around the diagonal (split), one where pos tags are used, but λ is not split around the diagonal (pos), one where an offset is used, but parameters aren't split about the diagonal (offset), one that's split about the diagonal and uses pos tags (pos & split) and finally one with all three (pos & split & offset). All are trained for 5 iterations, with uniform initialisation, where the first iteration only the translation probabilities are updated, and the other parameters are updated as well in the subsequent iterations. The same hyperparameters are

Model	Fr-En	Ar-En	Zh-En
Tokens	111M	46M	17.3M
(after)	110M	29.0M	10.4M
average	1.64	0.76	0.27
Model 4	15.5	6.3	2.2

Table 1: Token counts and average amount of time to train models (and separately training time for Model 4) on original corpora in one direction in hours, by corpus.

used as in (Dyer et al., 2013), with stepsize for updates to λ and γ during gradient ascent is 1000, and that for ω is 0.03, decaying after every gradient descent step by 0.9, using 8 steps every iteration. Both λ and γ are initialised to 6, and ω is initialised to 0. For these experiments the pos and pos & split use POS tags generated using the Stanford POS tagger (Toutanova and Manning, 2000), using the supplied models for all of the languages used in the experiments. For comparison, Model 4 is trained for 5 iterations using 5 iterations each of Model 1 and Model 3 as initialization, using GIZA++ (Och and Ney, 2003).

For the comparisons in AER, the corpora are used as-is, but for the BLEU comparisons, sentences longer than 50 words are filtered out. In Table 2 the sizes of the corpora before filtering are listed, as well as the time taken in hours to align the corpora for AER. As the training times for the different versions barely differ, only the average is displayed for the models here described and Model 4 training times are given for comparison. Note that the times for the models optimizing only λ and γ , and the model only optimizing ω still calculate the derivatives for the other parameters, and so could be made to be faster than here displayed. For both the BLEU and AER results, the alignments are generated in both directions, and symmetrised using the grow-diag-final-and heuristic, which in preliminary tests had shown to do best in terms of AER.

The results are given in Table 2. These scores were computed using the WMT2012 data as gold standard. The different extensions to the model make no difference to the AER scores for Chinese-English, and actually do slightly worse for French-English. In both cases, Model 4 does better than the models introduced here.

Model	Fr-En	Zh-En
Original	16.3	42.5
Split	16.8	42.5
Pos	16.6	42.5
Offset	16.8	42.5
Pos & Split	16.8	42.5
Pos & Split & Offset	16.7	42.5
Model 4	11.2	40.5

Table 2: AER results on Chinese-English and French-English data sets

Model	Fr-En	Ar-En	Zh-En
Original	25.9	43.8	32.8
Split	25.9	43.2	32.8
Pos	25.9	43.9	32.9
Offset	26.0	43.9	32.8
Pos & Split	26.0	44.1	33.2
Pos & Split & Offset	26.0	44.2	33.3
Model 4	26.8	43.9	32.4

Table 3: BLEU results on Chinese-English and French-English data sets

For the comparisons of translation quality, the models are trained up using a phrase-based translation system (Koehn et al., 2007) that used the above listed models to align the data. Language models were augmented with data outside of the corpora for Chinese-English (200M words total) and Arabic-English (100M words total). Test sets for Chinese are MT02, MT03, MT06 and MT08, for Arabic they were MT05, MT06 and MT08, and for French they were the newssyscom2009 data and the newstest 2009-2012 data.

The results are listed in Table 3¹. BLEU scores for Arabic-English and Chinese-English are computed with multiple references, while those for French-English are against a single reference. Although the different models made little difference in AER, there is quite a bit of variation in the BLEU scores between the different models. In all cases, the models conditioned on POS tags did better than the original model, by as much as 0.5 BLEU points. For Arabic-English as well as Chinese-English, the full model outperformed

¹The difference in these results compared to those reported in Dyer et al. (2013) is due to differences in corpus size, and the fact that a different translation model is used.

Model 4, in the case of Chinese-English by 0.9 BLEU points.

The low impact of the split and offset models are most likely due to the need to model all alignments in the corpus. The distributions can't skew too far to aligning to one direction, as that would lower the probability of a large amount of alignments. This is reflected in the resulting parameters λ , γ and ω that are estimated, as the first two do not differ much from the parameters estimated when both are kept the same, and the second tends to be very small.

As for the Pos model, it seems that only varying the symmetrical slope for the different POS tags doesn't capture the differences between distributions for POS tags. For example, the λ and γ parameters can differ quite a lot in the Pos & Split model when compared to the Pos model, with one side having a much smaller parameter and the other a much larger parameter for a given POS tag in the first model, and the single parameter being closer to the model average for the same POS tag in the second model.

The low variation in results between the different models for French-English might be explained by less word movement when translating between these languages, which could mean the original model is sufficient to capture this behaviour.

4 Conclusion

We have shown some extensions to a reparameterized IBM Model 2, allowing it to model word reordering better. Although these models don't improve on the baseline in terms of AER, they do better than the original in all three languages tested, and outperform M4 in two of these languages, at little cost in terms of training time. Future directions for this work include allowing for more expressivity of the alignment model by using a Beta distribution instead of the current exponential model.

5 Acknowledgments

Dr Cohn is the recipient of an Australian Research Council Future Fellowship (project number FT130101105).

References

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993.

The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19:263–311.

Chris Dyer, Victor Chahuneau, and Noah A Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of NAACL-HLT*, pages 644–648.

P. A. Fernandez, T. Foregger, and J. Pahikkala. 2006. Arithmetic-geometric series.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29:19–51, March.

Kristina Toutanova and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, EMNLP '00, pages 63–70, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dependency-based Pre-ordering for Chinese-English Machine Translation

Jingsheng Cai^{†*} Masao Utiyama[‡] Eiichiro Sumita[‡] Yujie Zhang[†]

[†]School of Computer and Information Technology, Beijing Jiaotong University

[‡]National Institute of Information and Communications Technology

joycetsai99@gmail.com

{mutiyama, eiichiro.sumita}@nict.go.jp

yjzhang@bjtu.edu.cn

Abstract

In statistical machine translation (SMT), syntax-based pre-ordering of the source language is an effective method for dealing with language pairs where there are great differences in their respective word orders. This paper introduces a novel pre-ordering approach based on dependency parsing for Chinese-English SMT. We present a set of dependency-based pre-ordering rules which improved the BLEU score by 1.61 on the NIST 2006 evaluation data. We also investigate the accuracy of the rule set by conducting human evaluations.

1 Introduction

SMT systems have difficulties translating between distant language pairs such as Chinese and English. The reason for this is that there are great differences in their word orders. Reordering therefore becomes a key issue in SMT systems between distant language pairs.

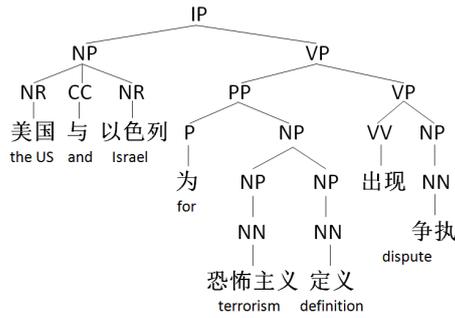
Previous work has shown that the approaches tackling the problem by introducing a pre-ordering procedure into phrase-based SMT (PBSMT) were effective. These pre-ordering approaches first parse the source language sentences to create parse trees. Then, syntactic reordering rules are applied to these parse trees with the goal of reordering the source language sentences into the word order of the target language. Syntax-based pre-ordering by employing constituent parsing have demonstrated effectiveness in many language pairs, such as English-French (Xia and McCord, 2004), German-English (Collins et al., 2005), Chinese-English (Wang et al., 2007; Zhang et al., 2008), and English-Japanese (Lee et al., 2010).

* This work was done when the first author was on an internship in NICT.

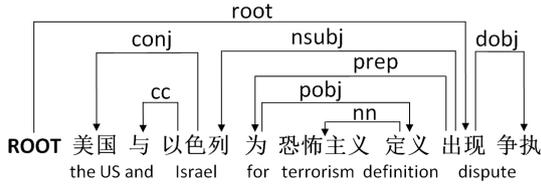
As a kind of constituent structure, HPSG (Pollard and Sag, 1994) parsing-based pre-ordering showed improvements in SVO-SOV translations, such as English-Japanese (Isozaki et al., 2010; Wu et al., 2011) and Chinese-Japanese (Han et al., 2012). Since dependency parsing is more concise than constituent parsing in describing sentences, some research has used dependency parsing in pre-ordering approaches for language pairs such as Arabic-English (Habash, 2007), and English-SOV languages (Xu et al., 2009; Katz-Brown et al., 2011). The pre-ordering rules can be made manually (Collins et al., 2005; Wang et al., 2007; Han et al., 2012) or extracted automatically from a parallel corpus (Xia and McCord, 2004; Habash, 2007; Zhang et al., 2007; Wu et al., 2011).

The purpose of this paper is to introduce a novel dependency-based pre-ordering approach through creating a pre-ordering rule set and applying it to the Chinese-English PBSMT system. Experiment results showed that our pre-ordering rule set improved the BLEU score on the NIST 2006 evaluation data by 1.61. Moreover, this rule set substantially decreased the total times of rule application about 60%, compared with a constituent-based approach (Wang et al., 2007). We also conducted human evaluations in order to assess its accuracy. To our knowledge, our manually created pre-ordering rule set is the first Chinese-English dependency-based pre-ordering rule set.

The most similar work to this paper is that of Wang et al. (2007). They created a set of pre-ordering rules for constituent parsers for Chinese-English PBSMT. In contrast, we propose a set of pre-ordering rules for dependency parsers. We argue that even though the rules by Wang et al. (2007) exist, it is almost impossible to automatically convert their rules into rules that are applicable to dependency parsers. In fact, we abandoned our initial attempts to automatically convert their rules into rules for dependency parsers, and



(a) A constituent parse tree



(b) Stanford typed dependency parse tree

Figure 1: A constituent parse tree and its corresponding Stanford typed dependency parse tree for the same Chinese sentence.

spent more than two months discovering the rules introduced in this paper. By applying our rules and Wang et al.’s rules, one can use both dependency and constituency parsers for pre-ordering in Chinese-English PBSMT.

This is especially important on the point of the system combination of PBSMT systems, because the diversity of outputs from machine translation systems is important for system combination (Cer et al., 2013). By using both our rules and Wang et al.’s rules, one can obtain diverse machine translation results because the pre-ordering results of these two rule sets are generally different.

Another similar work is that of (Xu et al., 2009). They created a pre-ordering rule set for dependency parsers from English to several SOV languages. In contrast, our rule set is for Chinese-English PBSMT. That is, the direction of translation is opposite. Because there are a lot of language specific decisions that reflect specific aspects of the source language and the language pair combination, our rule set provides a valuable resource for pre-ordering in Chinese-English PBSMT.

2 Dependency-based Pre-ordering Rule Set

Figure 1 shows a constituent parse tree and its Stanford typed dependency parse tree for the same

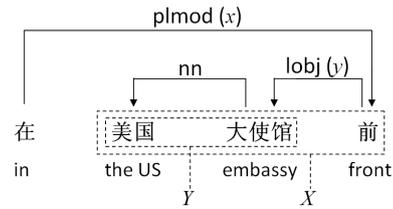


Figure 2: An example of a preposition phrase with a plmod structure. The phrase translates into “in front of the US embassy”.

Chinese sentence. As shown in the figure, the number of nodes in the dependency parse tree (i.e. 9) is much fewer than that in its corresponding constituent parse tree (i.e. 17). Because dependency parse trees are generally more concise than the constituent ones, they can conduct long-distance reorderings in a finer way. Thus, we attempted to conduct pre-ordering based on dependency parsing. There are two widely-used dependency systems – Stanford typed dependencies and CoNLL typed dependencies. For Chinese, there are 45 types of grammatical relations for Stanford typed dependencies (Chang et al., 2009) and 25 for CoNLL typed dependencies. As we thought that Stanford typed dependencies could describe language phenomena more meticulously owing to more types of grammatical relations, we preferred to use it for searching candidate pre-ordering rules.

We designed two types of formats in our dependency-based pre-ordering rules. They are:

Type-1: $x : y$

Type-2: $x - y$

Here, both x and y are *dependency relations* (e.g., plmod or lobj in Figure 2). We define the *dependency structure* of a dependency relation as the structure containing the dependent word (e.g., the word directly indicated by plmod, or “前” in Figure 2) and the whole subtree under the dependency relation (all of the words that directly or indirectly depend on the dependent word, or the words under “前” in Figure 2). Further, we define X and Y as the corresponding dependency structures of the dependency relations x and y , respectively. We define $X \setminus Y$ as structure X except Y . For example, in Figure 2, let x and y denote plmod and lobj dependency relations, then X represents “前” and all words under “前”, Y represents “大使馆” and all words under “大使馆”, and $X \setminus Y$ represents

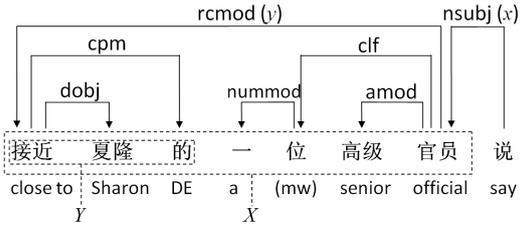


Figure 3: An example of rcmod structure within an nsubj structure. The phrase translates into “a senior official close to Sharon said”.

“前”. For Type-1, Y is a sub-structure of X . The rule repositions $X \setminus Y$ to the position before Y . For Type-2, X and Y are ordered sibling structures under a same parent node. The rule repositions X to the position after Y .

We obtained rules as the following steps:

- 1 Search the Chinese dependency parse trees in the corpus and rank all of the structures matching the two types of rules respectively according to their frequencies. Note that while calculating the frequencies of Type-1 structures, we dismissed the structures in which X occurred before Y originally.
- 2 Filtration. 1) Filter out the structures which occurred less than 5,000 times. 2) Filter out the structures from which it was almost impossible to derive candidate pre-ordering rules because x or y was an “irrespective” dependency relation, for example, root, conj, cc and so on.
- 3 Investigate the remaining structures. For each kind of structure, we selected some of the sample dependency parse trees that contained it, tried to restructure the parse trees according to the matched rule and judged the re-ordered Chinese phrases. If the reordering produced a Chinese phrase that had a closer word order to that of the English one, this structure would be a candidate pre-ordering rule.
- 4 Conduct primary experiments which used the same training set and development set as the experiments described in Section 3. In the primary experiments, we tested the effectiveness of the candidate rules and filtered the ones that did not work based on the BLEU scores on the development set.

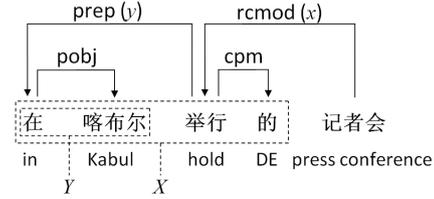


Figure 4: An example of rcmod structure with a preposition modifier. The phrase translates into “a press conference held in Kabul”.

As a result, we obtained eight pre-ordering rules in total, which can be divided into three dependency relation categories. They are: plmod (localizer modifier of a preposition), rcmod (relative clause modifier) and prep (preposition modifier). Each of these categories are discussed in detail below.

plmod Figure 2 shows an example of a prepositional phrase with a plmod structure, which translates literally into “in the US embassy front”. In Chinese, the dependent word of a plmod relation (e.g., “前” in Figure 2) occurs in the last position of the prepositional phrase. However, in English, this kind of word (e.g., “front” in the caption of Figure 2) always occur directly after prepositions, which is to say, in the second position in a prepositional phrase. Therefore, we applied a rule **plmod : lobj** (localizer object) to reposition the dependent word of the plmod relation (e.g., “前” in Figure 2) to the position before the lobj structure (e.g., “美国 大使馆” in Figure 2). In this case, it also comes directly after the preposition. Similarly, we created a rule **plmod : lcomp** (clausal complement of a localizer).

rcmod Figure 3 shows an example of an rcmod structure under an nsubj (nominal subject) structure. Here “mw” means “measure word”. As shown in the figure, relative clause modifiers in Chinese (e.g., “接近夏隆的” in Figure 3) occurs before the noun being modified, which is in contrast to English (e.g., “close to Sharon” in the caption of Figure 3), where they come after. Thus, we introduced a series of rules **NOUN : rcmod** to restructure rcmod structures so that the noun is moved to the head. In this example, with the application of an **nsubj : rcmod** rule, the phrase can be translated into “a senior official close to Sharon say”, which has a word order very close to English. Since a noun can be nsubj, dobj (direct object), pobj (prepositional object) and lobj

Type	System	Parser	BLEU	Counts	#Sent.
-	No pre-ordering	-	29.96	-	-
Constituent	WR07	Berkeley	31.45	2,561,937	852,052
Dependency	OUR DEP 1	Berkeley Const.	31.54	978,013	556,752
	OUR DEP 2	Mate	31.57	947,441	547,084

Table 1: The comparison of four systems, including the performance (BLEU) on the test set, the total count of each rule set and the number of sentences they were applied to on the training set.

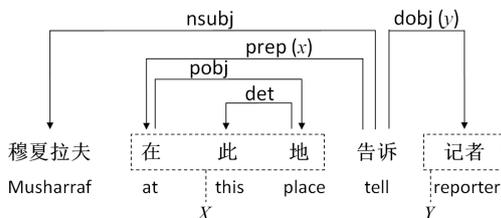


Figure 5: An example of verb phrase with a preposition modifier. The phrase translates into “Musharraf told reporters here”.

in Stanford typed dependencies, we created four rules from the NOUN pattern. Note that for some preposition modifiers, we needed a rule **rcmod : prep** to conduct the same work. For instance, the Chinese phrase in Figure 4 can be translated into “hold in Kabul press conference” with the application of this rule.

prep Within verb phrases, the positions of prep structures are quite different between Chinese and English. Figure 5 shows an example of a verb phrase with a preposition modifier (prep), which literally translates into “Musharraf at this place tell reporter”. Recognizing that prep structures occur before the verb in Chinese (e.g., “在此地” in Figure 5) but after the verb in English (usually in the last position of a verb phrase, e.g., “here” in the caption of Figure 5), we applied a rule **prep - dobj** to reposition prep structures after their sibling dobj structures.

In summary, the dependency-based pre-ordering rule set has eight rules: **plmod : lobj**, **plmod : lcomp**, **nsbj : rcmod**, **dobj : rcmod**, **pobj : rcmod**, **lobj : rcmod**, **rcmod : prep**, and **prep - dobj**.

3 Experiments

We used the MOSES PBSMT system (Koehn et al., 2007) in our experiments. The training data, which included those data used in Wang et al. (2007), contained 1 million pairs of sentences ex-

tracted from the Linguistic Data Consortium’s parallel news corpora. Our development set was the official NIST MT evaluation data from 2002 to 2005, consisting of 4476 Chinese-English sentences pairs. Our test set was the NIST 2006 MT evaluation data, consisting of 1664 sentence pairs. We employed the Stanford Segmenter¹ to segment all of the data sets. For evaluation, we used BLEU scores (Papineni et al., 2002).

We implemented the constituent-based pre-ordering rule set in Wang et al. (2007) for comparison, which is called WR07 below. The Berkeley Parser (Petrov et al., 2006) was employed for parsing the Chinese sentences. For training the Berkeley Parser, we used Chinese Treebank (CTB) 7.0.

We conducted our dependency-based pre-ordering experiments on the Berkeley Parser and the Mate Parser (Bohnet, 2010), which were shown to be the two best parsers for Stanford typed dependencies (Che et al., 2012). First, we converted the constituent parse trees in the results of the Berkeley Parser into dependency parse trees by employing a tool in the Stanford Parser (Klein and Manning, 2003). For the Mate Parser, POS tagged inputs are required both in training and in inference. Thus, we then extracted the POS information from the results of the Berkeley Parser and used these as the pre-specified POS tags for the Mate Parser. Finally, we applied our dependency-based pre-ordering rule set to the dependency parse trees created from the converted Berkeley Parser and the Mate Parser, respectively.

Table 1 presents a comparison of the system without pre-ordering, the constituent system using WR07 and two dependency systems employing the converted Berkeley Parser and the Mate Parser, respectively. It shows the BLEU scores on the test set and the statistics of pre-ordering on the training set, which includes the total count of each rule set and the number of sentences they were ap-

¹<http://nlp.stanford.edu/software/segmenter.shtml>

Category	Count	Correct	Incorrect	Accuracy
plmod	42	26	16	61.9%
rcmod	89	49	40	55.1%
prep	54	36	18	66.7%
All	185	111	74	60.0%

Table 2: Accuracy of the dependency-based pre-ordering rules on a set of 200 sentences randomly selected from the development set.

plied to. Both of our dependency systems outperformed WR07 slightly but were not significant at $p = 0.05$. However, both of them substantially decreased the total times about 60% (or 1,600,000) for pre-ordering rule applications on the training set, compared with WR07. In our opinion, the reason for the great decrease was that the dependency parse trees were more concise than the constituent parse trees in describing sentences and they could also describe the reordering at the sentence level in a finer way. In contrast, the constituent parse trees were more redundant and they needed more nodes to conduct long-distance reordering. In this case, the affect of the performance of the constituent parsers on pre-ordering is larger than that of the dependency ones so that the constituent parsers are likely to bring about more incorrect pre-orderings.

Similar to Wang et al. (2007), we carried out human evaluations to assess the accuracy of our dependency-based pre-ordering rules by employing the system “OUR DEP 2” in Table 1. The evaluation set contained 200 sentences randomly selected from the development set. Among them, 107 sentences contained at least one rule and the rules were applied 185 times totally. Since the accuracy check for dependency parse trees took great deal of time, we did not try to select error free (100% accurately parsed) sentences. A bilingual speaker of Chinese and English looked at an original Chinese phrase and the pre-ordered one with their corresponding English phrase and judged whether the pre-ordering obtained a Chinese phrase that had a closer word order to the English one. Table 2 shows the accuracies of three categories of our dependency-based pre-ordering rules. The overall accuracy of this rule set is 60.0%, which is almost at the same level as the WR07 rule set (62.1%), according to the similar evaluation (200 sentences and one annotator) conducted in Wang et al. (2007). Notice that some of the incorrect pre-orderings may be caused by erroneous parsing as also suggested by Wang et

al. (2007). Through human evaluations, we found that 19 out of the total 74 incorrect pre-orderings resulted from errors in parsing. Among them, 13 incorrect pre-orderings applied the rules of the rcmod category. The analysis suggests that we need to introduce constraints on the rule application of this category in the future.

4 Conclusion

In this paper, we introduced a novel pre-ordering approach based on dependency parsing for a Chinese-English PBSMT system. The results showed that our approach achieved a BLEU score gain of 1.61. Moreover, our dependency-based pre-ordering rule set substantially decreased the time for applying pre-ordering rules about 60% compared with WR07, on the training set of 1M sentences pairs. The overall accuracy of our rule set is 60.0%, which is almost at the same level as the WR07 rule set. These results indicated that dependency parsing is more effective for conducting pre-ordering for Chinese-English PBSMT. Although our work focused on Chinese, the ideas can also be applied to other languages.

In the future, we attempt to create more efficient pre-ordering rules by exploiting the rich information in dependency structures.

Acknowledgments

We thank the anonymous reviewers for their valuable comments and suggestions. This work is supported in part by the International Science & Technology Cooperation Program of China (Grant No. 2014DFA11350) and Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China.

References

Bernd Bohnet. 2010. Very high accuracy and fast dependency parsing is not a contradiction. In *Proceed-*

- ings of the 23rd International Conference on Computational Linguistics (COLING 2010).
- Daniel Cer, Christopher D. Manning, and Dan Jurafsky. 2013. Positive Diversity Tuning for Machine Translation System Combination. In *Proceedings of the Eighth Workshop on Statistical Machine Translation (WMT 2013)*.
- Pi-Chuan Chang, Huihsin Tseng, Dan Jurafsky, and Christopher D. Manning. 2009. Discriminative reordering with Chinese grammatical relations features. In *Proceedings of the HLT-NAACL Workshop on Syntax and Structure in Statistical Translation*, pages 51-59.
- Wanxiang Che, Valentin Spitzkovsky, and Ting Liu. 2012. A comparison of Chinese parsers for Stanford dependencies. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 11-16.
- Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 531-540.
- Dan Han, Katsuhito Sudoh, Xianchao Wu, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. 2012. Head Finalization reordering for Chinese-to-Japanese machine translation. In *Proceedings of SSST-6, Sixth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 57-66.
- Nizar Habash. 2007. Syntactic preprocessing for statistical machine translation. In *Proceedings of the 11th Machine Translation Summit (MT-Summit)*.
- Hideki Isozaki, Katsuhito Sudoh, Hajime Tsukada, and Kevin Duh. 2010. Head Finalization: A simple reordering rule for SOV languages. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 250-257.
- Jason Katz-Brown, Slav Petrov, Ryan McDonald, Franz J. Och, David Talbot, Hiroshi Ichikawa, Masakazu Seno, and Hideto Kazawa. 2011. Training a parser for machine translation reordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 183-192.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 423-430.
- 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 *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177-180.
- Young-Suk Lee, Bing Zhao, and Xiaoqian Luo. 2010. Constituent reordering and syntax models for English-to-Japanese statistical machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 626-634.
- 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.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433-440.
- Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press.
- Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 737-745.
- Xianchao Wu, Katsuhito Sudoh, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. 2011. Extracting preordering rules from predicate-argument structures. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 29-37.
- Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of Coling 2004*, pages 508-514.
- Peng Xu, Jaeho Kang, Michael Ringgaard, and Franz J. Och. 2009. Using a dependency parser to improve SMT for subject-object-verb languages. In *Proceedings of HLT-NAACL*, pages 245-253.
- Jiajun Zhang, Chengqing Zong, and Shoushan Li. 2008. Sentence type based reordering model for statistical machine translation. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 1089-1096.
- Yuqi Zhang, Richard Zens, and Hermann Ney. 2011. Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In *HLT-NAACL Workshop on Syntax and Structure in Statistical Translation*, pages 1-8.

Generalized Character-Level Spelling Error Correction

Noura Farra, Nadi Tomeh[†], Alla Rozovskaya, Nizar Habash

Center for Computational Learning Systems, Columbia University

{noura, alla, habash}@cccls.columbia.edu

[†]LIPN, Université Paris 13, Sorbonne Paris Cité

nadi.tomeh@lipn.univ-paris13.fr

Abstract

We present a generalized discriminative model for spelling error correction which targets character-level transformations. While operating at the character level, the model makes use of word-level and contextual information. In contrast to previous work, the proposed approach learns to correct a variety of error types without guidance of manually-selected constraints or language-specific features. We apply the model to correct errors in Egyptian Arabic dialect text, achieving 65% reduction in word error rate over the input baseline, and improving over the earlier state-of-the-art system.

1 Introduction

Spelling error correction is a longstanding Natural Language Processing (NLP) problem, and it has recently become especially relevant because of the many potential applications to the large amount of informal and unedited text generated online, including web forums, tweets, blogs, and email. Misspellings in such text can lead to increased sparsity and errors, posing a challenge for many NLP applications such as text summarization, sentiment analysis and machine translation.

In this work, we present GSEC, a Generalized character-level Spelling Error Correction model, which uses supervised learning to map input characters into output characters in context. The approach has the following characteristics:

Character-level Corrections are learned at the character-level¹ using a supervised sequence labeling approach.

Generalized The input space consists of all characters, and a single classifier is used to learn

common error patterns over all the training data, without guidance of specific rules.

Context-sensitive The model looks beyond the context of the current word, when making a decision at the character-level.

Discriminative The model provides the freedom of adding a number of different features, which may or may not be language-specific.

Language-Independent In this work, we integrate only language-independent features, and therefore do not consider morphological or linguistic features. However, we apply the model to correct errors in Egyptian Arabic dialect text, following a conventional orthography standard, CODA (Habash et al., 2012).

Using the described approach, we demonstrate a word-error-rate (WER) reduction of 65% over a do-nothing input baseline, and we improve over a state-of-the-art system (Eskander et al., 2013) which relies heavily on language-specific and manually-selected constraints. We present a detailed analysis of mistakes and demonstrate that the proposed model indeed learns to correct a wider variety of errors.

2 Related Work

Most earlier work on automatic error correction addressed spelling errors in English and built models of correct usage on native English data (Kukich, 1992; Golding and Roth, 1999; Carlson and Fette, 2007; Banko and Brill, 2001). Arabic spelling correction has also received considerable interest (Ben Othmane Zribi and Ben Ahmed, 2003; Haddad and Yaseen, 2007; Hassan et al., 2008; Shaalan et al., 2010; Alkanhal et al., 2012; Eskander et al., 2013; Zaghouni et al., 2014).

Supervised spelling correction approaches trained on paired examples of errors and their corrections have recently been applied for non-native English correction (van Delden et al., 2004; Li et al., 2012; Gamon, 2010; Dahlmeier and Ng, 2012;

¹We use the term ‘character’ strictly in the alphabetic sense, not the logographic sense (as in the Chinese script).

Rozovskaya and Roth, 2011). Discriminative models have been proposed at the word-level for error correction (Duan et al., 2012) and for error detection (Habash and Roth, 2011).

In addition, there has been growing work on lexical normalization of social media data, a somewhat related problem to that considered in this paper (Han and Baldwin, 2011; Han et al., 2013; Subramaniam et al., 2009; Ling et al., 2013).

The work of Eskander et al. (2013) is the most relevant to the present study: it presents a character-edit classification model (CEC) using the same dataset we use in this paper.² Eskander et al. (2013) analyzed the data to identify the seven most common types of errors. They developed seven classifiers and applied them to the data in succession. This makes the approach tailored to the specific data set in use and limited to a specific set of errors. In this work, a single model is considered for all types of errors. The model considers every character in the input text for a possible spelling error, as opposed to looking only at certain input characters and contexts in which they appear. Moreover, in contrast to Eskander et al. (2013), it looks beyond the boundary of the current word.

3 The GSEC Approach

3.1 Modeling Spelling Correction at the Character Level

We recast the problem of spelling correction into a sequence labeling problem, where for each input character, we predict an **action label** describing how to transform it to obtain the correct character. The proposed model therefore transforms a given input sentence $\mathbf{e} = e_1, \dots, e_n$ of n characters that possibly include errors, to a corrected sentence \mathbf{c} of m characters, where corrected characters are produced by one of the following four actions applied to each input character e_i :

- *ok*: e_i is passed without transformation.
- *substitute – with(c)*: e_i is substituted with a character c where c could be any character encountered in the training data.
- *delete*: e_i is deleted.
- *insert(c)*: A character c is inserted *before* e_i . To address errors occurring at the end

²Eskander et al. (2013) also considered a slower, more expensive, and more language-specific method using a morphological tagger (Habash et al., 2013) that outperformed the CEC model; however, we do not compare to it in this paper.

Input	Action Label
<i>k</i>	<i>substitute-with(c)</i>
<i>o</i>	<i>ok</i>
<i>r</i>	<i>insert(r)</i>
<i>e</i>	<i>ok</i>
<i>c</i>	<i>ok</i>
<i>t</i>	<i>ok</i>
<i>d</i>	<i>delete</i>

Table 1: Character-level spelling error correction process on the input word *korectd*, with the reference word *correct*

	Train	Dev	Test
Sentences	10.3K	1.67K	1.73K
Characters	675K	106K	103K
Words	134K	21.1K	20.6K

Table 2: ARZ Egyptian dialect corpus statistics

of the sentence, we assume the presence of a dummy sentence-final *stop* character.

We use a multi-class SVM classifier to predict the action labels for each input character $e_i \in \mathbf{e}$. A decoding process is then applied to transform the input characters accordingly to produce the corrected sentence. Note that we consider the space character as a character like any other, which gives us the ability to correct word merge errors with space character insertion actions and word split errors with space character deletion actions. Table 1 shows an example of the spelling correction process.

In this paper, we only model single-edit actions and ignore cases where a character requires multiple edits (henceforth, **complex** actions), such as multiple insertions or a combination of insertions and substitutions. This choice was motivated by the need to reduce the number of output labels, as many infrequent labels are generated by complex actions. An error analysis of the training data, described in detail in section 3.2, showed that complex errors are relatively infrequent (4% of data). We plan to address these errors in future work.

Finally, in order to generate the training data in the described form, we require a parallel corpus of erroneous and corrected reference text (described below), which we align at the character level. We use the alignment tool **ScLite** (Fiscus, 1998), which is part of the SCTL Toolkit.

3.2 Description of Data

We apply our model to correcting Egyptian Arabic dialect text. Since there is no standard dialect orthography adopted by native speakers of Arabic dialects, it is common to encounter multiple

Action	% Errors	Example Error ⇒ Reference
Substitute	80.9	
^E <i>Alif A</i> forms ($\tilde{V}\tilde{W}\tilde{A}\tilde{A}/\tilde{A}/\tilde{A}$)	33.3	<i>AHdhm</i> ⇒ <i>ĀHdhm</i> أحدهم ⇒ أحدهم
^E <i>Ya</i> ي/ي forms (y/\tilde{y})	26.7	<i>ϰly</i> ⇒ <i>ϰly</i> علي ⇒ على
^E <i>h/h̄</i> ه/ه, <i>h/w</i> و/و forms	14.9	<i>kfrh</i> ⇒ <i>kfrh</i> كفرة ⇒ كفرة
^E <i>h/H</i> ه/ح forms	2.2	<i>htϰmlhA</i> ⇒ <i>HtϰmlhA</i> هتعملها ⇒ هتعملها
Other substitutions	3.8	<i>AltAnyh̄</i> ⇒ <i>AlθAnyh̄</i> الثانية ⇒ الثانية ; <i>dA</i> ⇒ <i>dh</i> دا ⇒ ده
Insert	10.5	
^{EP} <i>Insert {A}</i>	3.0	<i>ktbw</i> ⇒ <i>ktbwA</i> كتبوا ⇒ كتبوا
^{EP} <i>Insert {space}</i>	2.9	<i>mAtzϰlš</i> ⇒ <i>mA tzϰlš</i> ما تزعلش ⇒ ما تزعلش
Other insertion actions	4.4	<i>Aly</i> ⇒ <i>Ally</i> الي ⇒ الي
Delete	4.7	
^E <i>Del{A}</i>	2.4	<i>whmA</i> ⇒ <i>whm</i> وهم ⇒ وهم
Other deletion actions	2.3	<i>wfyh</i> ⇒ <i>wfy</i> وفيه ⇒ وفي
Complex	4.0	<i>mykwnš</i> ⇒ <i>mA ykwnš</i> ما يكونش ⇒ ميكونش

Table 3: Character-level distribution of correction labels. We model all types of transformations except *complex* actions, and rare *Insert* labels with counts below a tuned threshold. The *Delete* label is a single label that comprises all deletion actions. Labels modeled by Eskander et al. (2013) are marked with ^E, and ^{EP} for cases modeled partially, for example, the *Insert{A}* would only be applied at certain positions such as the end of the word.

spellings of the same word. The CODA orthography was proposed by Habash et al. (2012) in an attempt to standardize dialectal writing, and we use it as a reference of correct text for spelling correction following the previous work by Eskander et al. (2013). We use the same corpus (labeled "ARZ") and experimental setup splits used by them. The ARZ corpus was developed by the Linguistic Data Consortium (Maamouri et al., 2012a-e). See Table 2 for corpus statistics.

Error Distribution Table 3 presents the distribution of correction action labels that correspond to spelling errors in the training data together with examples of these errors.³ We group the actions into: *Substitute*, *Insert*, *Delete*, and *Complex*, and also list common transformations within each group. We further distinguish between the phenomena modeled by our system and by Eskander et al. (2013). At least 10% of all generated action labels are not handled by Eskander et al. (2013).

3.3 Features

Each input character is represented by a feature vector. We include a set of basic features inspired by Eskander et al. (2013) in their CEC system and additional features for further improvement.

Basic features We use a set of nine basic features: the given character, the preceding and following two characters, and the first two and last

³Arabic transliteration is presented in the Habash-Soudi-Buckwalter scheme (Habash et al., 2007). For more information on Arabic orthography in NLP, see (Habash, 2010).

two characters in the word. These are the same features used by CEC, except that CEC does not include characters beyond the word boundary, while we consider space characters as well as characters from the previous and next words.

Ngram features We extract sequences of characters corresponding to the current character and the following and previous two, three, or four characters. We refer to these sequences as bigrams, trigrams, or 4-grams, respectively. These are an extension of the basic features and allow the model to look beyond the context of the current word.

3.4 Maximum Likelihood Estimate (MLE)

We implemented another approach for error correction based on a word-level maximum likelihood model. The MLE method uses a unigram model which replaces each input word with its most likely correct word based on counts from the training data. The intuition behind MLE is that it can easily correct frequent errors; however, it is quite dependent on the training data.

4 Experiments

4.1 Model Evaluation

Setup The training data was extracted to generate the form described in Section 3.1, using the Sclite tool (Fiscus, 1998) to align the input and reference sentences. A speech effect handling step was applied as a preprocessing step to all models.

This step removes redundant repetitions of characters in sequence, e.g., كتييييير *ktyyyyyr* ‘veeeery’. The same speech effect handling was applied by Eskander et al. (2013).

For classification, we used the SVM implementation in YamCha (Kudo and Matsumoto, 2001), and trained with different variations of the features described above. Default parameters were selected for training ($c=1$, quadratic kernel, and context window of ± 2).

In all results listed below, the baseline corresponds to the do-nothing baseline of the input text.

Metrics Three evaluation metrics are used. The word-error-rate **WER** metric is computed by summing the total number of word-level substitution errors, insertion errors, and deletion errors in the output, and dividing by the number of words in the **reference**. The correct-rate **Corr** metric is computed by dividing the number of correct output words by the total number of words in the reference. These two metrics are produced by Sclite (Fiscus, 1998), using automatic alignment. Finally, the accuracy **Acc** metric, used by Eskander et al. (2013), is a simple string matching metric which enforces a word alignment that pairs words in the reference to those of the output. It is calculated by dividing the number of correct output words by the number of words in the **input**. This metric assumes no split errors in the data (a word incorrectly split into two words), which is the case in the data we are working with.

Character-level Model Evaluation The performance of the generalized spelling correction model (GSEC) on the dev data is presented in the first half of Table 4. The results of the Eskander et al. (2013) CEC system are also presented for the purpose of comparison. We can see that using a single classifier, the generalized model is able to outperform CEC, which relies on a cascade of classifiers ($p = 0.03$ for the basic model and $p < 0.0001$ for the best model, GSEC+4grams).⁴

Model Combination Evaluation Here we present results on combining GSEC with the MLE component (GSEC+MLE). We combine the two models in cascade: the MLE component is applied to the output of GSEC. To train the MLE model, we use the word pairs obtained from the original training data, rather than from the output of GSEC. We found that this configuration allows

Approach	Corr%/WER	Acc%
Baseline	75.9/24.2	76.8
CEC	88.7/11.4	90.0
GSEC	89.7/10.4*	90.3*
GSEC+2grams	90.6/9.5*	91.2*
GSEC+4grams	91.0/9.2*	91.6*
MLE	89.7/10.4	90.5
CEC + MLE	90.8/9.4	91.5
GSEC+MLE	91.0/9.2	91.3
GSEC+4grams+ MLE	91.7/8.3*	92.2*

Table 4: Model Evaluation. GSEC represents the generalized character-level model. CEC represents the character-level-edit classification model of Eskander et al. (2013). Rows marked with an asterisk (*) are statistically significant compared to CEC (for the first half of the table) or CEC+MLE (for the second half of the table), with $p < 0.05$.

us to include a larger sample of word pair errors for learning, because our model corrects many errors, leaving fewer example pairs to train an MLE post-processor. The results are shown in the second half of Table 4.

We first observe that MLE improves the performance of both CEC and GSEC. In fact, CEC+MLE and GSEC+MLE perform similarly ($p = 0.36$, not statistically significant). When adding features that go beyond the word boundary, we achieve an improvement over MLE, GSEC+MLE, and CEC+MLE, all of which are mostly restricted within the boundary of the word. The best GSEC model outperforms CEC+MLE ($p < 0.0001$), achieving a **WER** of 8.3%, corresponding to 65% reduction compared to the baseline. It is worth noting that adding the MLE component allows Eskander’s CEC to recover various types of errors that were not modeled previously. However, the contribution of MLE is limited to words that are in the training data. On the other hand, because GSEC is trained on character transformations, it is likely to generalize better to words unseen in the training data.

Results on Test Data Table 5 presents the results of our best model (GSEC+4grams), and best model+MLE. The latter achieves a 92.1% **Acc** score. The **Acc** score reported by Eskander et al. (2013) for CEC+MLE is 91.3%. The two results are statistically significant ($p < 0.0001$) with respect to CEC and CEC+MLE respectively.

Approach	Corr%/WER	Acc%
Baseline	74.5/25.5	75.5
GSEC+4grams	90.9/9.1	91.5
GSEC+4grams+ MLE	91.8/8.3	92.1

Table 5: Evaluation on test data.

⁴Significance results are obtained using McNemar’s test.

4.2 Error Analysis

To gain a better understanding of the performance of the models on different types of errors and their interaction with the MLE component, we separate the words in the dev data into: (1) words seen in the training data, or in-vocabulary words (**IV**), and (2) out-of-vocabulary (**OOV**) words not seen in the training data. Because the MLE model maps every input word to its most likely gold word seen in the training data, we expect the MLE component to recover a large portion of errors in the IV category (but not all, since an input word can have multiple correct readings depending on the context). On the other hand, the recovery of errors in OOV words indicates how well the character-level model is doing independently of the MLE component. Table 6 presents the performance, using the **Acc** metric, on each of these types of words. Here our best model (GSEC+4grams) is considered.

	#Inp Words	Baseline	CEC+MLE	GSEC+MLE
OOV	3,289 (17.2%)	70.7	76.5	80.5
IV	15,832 (82.8%)	78.6	94.6	94.6
Total	19,121 (100%)	77.2	91.5	92.2

Table 6: Accuracy of character-level models shown separately on out-of-vocabulary and in-vocabulary words.

When considering words seen in the training data, CEC and GSEC have the same performance. However, when considering OOV words, GSEC performs significantly better ($p < 0.0001$), verifying our hypothesis that a generalized model reduces dependency on training data. The data is heavily skewed towards IV words (83%), which explains the generally high performance of MLE.

We performed a manual error analysis on a sample of 50 word errors from the IV set and found that all of the errors came from gold annotation errors and inconsistencies, either in the dev or train. We then divided the character transformations in the OOV words into four groups: (1) characters that were unchanged by the gold (**X-X** transformations), (2) character transformations modeled by CEC (**X-Y CEC**), (3) character transformations not modeled by CEC, and which include all phenomena that were only partially modeled by CEC (**X-Y not CEC**), and (4) complex errors. The character-level accuracy on each of these groups is shown in Table 7.

Both CEC and GSEC do much better on the second group of character transformations (that is, **X-Y CEC**) than on the third group (**X-Y not CEC**). This is not surprising because the former

Type	#Chars	Example	CEC	GSEC
X-X	16502	<i>m-m, space-space</i>	99.25	99.33
X-Y (<i>CEC</i>)	609	<i>ħ-h, h-ħ, Ā-A</i> <i>A-Ā, y-ỵ</i>	80.62	83.09
X-Y (<i>not CEC</i>)	161	<i>t-θ, del{w}</i> <i>n-ins{space}</i>	31.68	43.48
Complex	32	<i>n-ins{A}{m}</i>	37.5	15.63

Table 7: Character-level accuracy on different transformation types for out-of-vocabulary words. For complex transformations, the accuracy represents the complex category recognition rate, and not the actual correction accuracy.

transformations correspond to phenomena that are most common in the training data. For GSEC, they are learned automatically, while for CEC they are selected and modeled explicitly. Despite this fact, GSEC generalizes better to OOV words. As for the third group, both CEC and GSEC perform more poorly, but GSEC corrects more errors (43.48% vs. 31.68% accuracy). Finally, CEC is better at recognizing complex errors, which, although are not modeled explicitly by CEC, can sometimes be corrected as a result of applying multiple classifiers in cascade. Dealing with complex errors, though there are few of them in this dataset, is an important direction for future work, and for generalizing to other datasets, e.g., (Zaghoulani et al., 2014).

5 Conclusions

We showed that a generalized character-level spelling error correction model can improve spelling error correction on Egyptian Arabic data. This model learns common spelling error patterns automatically, without guidance of manually selected or language-specific constraints. We also demonstrate that the model outperforms existing methods, especially on out-of-vocabulary words.

In the future, we plan to extend the model to use word-level language models to select between top character predictions in the output. We also plan to apply the model to different datasets and different languages. Finally, we plan to experiment with more features that can also be tailored to specific languages by using morphological and linguistic information, which was not explored in this paper.

Acknowledgments

This publication was made possible by grant NPRP-4-1058-1-168 from the Qatar National Research Fund (a member of the Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

- Mohamed I. Alkanhal, Mohammed A. Al-Badrashiny, Mansour M. Alghamdi, and Abdulaziz O. Al-Qabbany. 2012. Automatic Stochastic Arabic Spelling Correction With Emphasis on Space Insertions and Deletions. *IEEE Transactions on Audio, Speech & Language Processing*, 20:2111–2122.
- Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, pages 26–33, Toulouse, France, July.
- Chiraz Ben Othmane Zribi and Mohammed Ben Ahmed. 2003. Efficient Automatic Correction of Misspelled Arabic Words Based on Contextual Information. In *Proceedings of the Knowledge-Based Intelligent Information and Engineering Systems Conference*, Oxford, UK.
- Andrew Carlson and Ian Fette. 2007. Memory-based context-sensitive spelling correction at web scale. In *Proceedings of the IEEE International Conference on Machine Learning and Applications (ICMLA)*.
- Daniel Dahlmeier and Hwee Tou Ng. 2012. A beam-search decoder for grammatical error correction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 568–578.
- Huizhong Duan, Yanen Li, ChengXiang Zhai, and Dan Roth. 2012. A discriminative model for query spelling correction with latent structural svm. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1511–1521, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ramy Eskander, Nizar Habash, Owen Rambow, and Nadi Tomeh. 2013. Processing spontaneous orthography. In *The Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '13.
- Jon Fiscus. 1998. ScLite scoring package version 1.5. US National Institute of Standard Technology (NIST), URL <http://www.itl.nist.gov/iaui/894.01/tools>.
- Michael Gamon. 2010. Using mostly native data to correct errors in learners' writing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 163–171, Los Angeles, California, June.
- Andrew R. Golding and Dan Roth. 1999. A Winnow based approach to context-sensitive spelling correction. *Machine Learning*, 34(1-3):107–130.
- Nizar Habash and Ryan M. Roth. 2011. Using deep morphology to improve automatic error detection in arabic handwriting recognition. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 875–884, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Nizar Habash, Abdelhadi Souidi, and Tim Buckwalter. 2007. On Arabic Transliteration. In A. van den Bosch and A. Souidi, editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Springer.
- Nizar Habash, Mona Diab, and Owen Rambow. 2012. Conventional orthography for dialectal Arabic. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may. European Language Resources Association (ELRA).
- Nizar Habash, Ryan Roth, Owen Rambow, Ramy Eskander, and Nadi Tomeh. 2013. Morphological Analysis and Disambiguation for Dialectal Arabic. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Atlanta, GA.
- Nizar Habash. 2010. *Introduction to Arabic Natural Language Processing*. Morgan & Claypool Publishers.
- Bassam Haddad and Mustafa Yaseen. 2007. Detection and Correction of Non-Words in Arabic: A Hybrid Approach. *International Journal of Computer Processing Of Languages (IJCPOL)*.
- Bo Han and Timothy Baldwin. 2011. Lexical normalization of short text messages: Makn sens a# twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 368–378. Association for Computational Linguistics.
- Bo Han, Paul Cook, and Timothy Baldwin. 2013. Lexical normalization for social media text. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(1):5.
- Ahmed Hassan, Sara Noeman, and Hany Hassan. 2008. Language Independent Text Correction using Finite State Automata. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP 2008)*.
- Taku Kudo and Yuji Matsumoto. 2001. Chunking with support vector machines. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, NAACL '01, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Karen Kukich. 1992. Techniques for Automatically Correcting Words in Text. *ACM Computing Surveys*, 24(4).

- Yanen Li, Huizhong Duan, and ChengXiang Zhai. 2012. A generalized hidden markov model with discriminative training for query spelling correction. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval, SIGIR '12*, pages 611–620, New York, NY, USA. ACM.
- Wang Ling, Chris Dyer, Alan W Black, and Isabel Trancoso. 2013. Paraphrasing 4 microblog normalization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 73–84, Seattle, Washington, USA, October. Association for Computational Linguistics.
- Mohamed Maamouri, Ann Bies, Seth Kulick, Sondos Krouna, Dalila Tabassi, and Michael Ciul. 2012a. Egyptian Arabic Treebank DF Part 1 V2.0. LDC catalog number LDC2012E93.
- Mohamed Maamouri, Ann Bies, Seth Kulick, Sondos Krouna, Dalila Tabassi, and Michael Ciul. 2012b. Egyptian Arabic Treebank DF Part 2 V2.0. LDC catalog number LDC2012E98.
- Mohamed Maamouri, Ann Bies, Seth Kulick, Sondos Krouna, Dalila Tabassi, and Michael Ciul. 2012c. Egyptian Arabic Treebank DF Part 3 V2.0. LDC catalog number LDC2012E89.
- Mohamed Maamouri, Ann Bies, Seth Kulick, Sondos Krouna, Dalila Tabassi, and Michael Ciul. 2012d. Egyptian Arabic Treebank DF Part 4 V2.0. LDC catalog number LDC2012E99.
- Mohamed Maamouri, Ann Bies, Seth Kulick, Sondos Krouna, Dalila Tabassi, and Michael Ciul. 2012e. Egyptian Arabic Treebank DF Part 5 V2.0. LDC catalog number LDC2012E107.
- Alla Rozovskaya and Dan Roth. 2011. Algorithm selection and model adaptation for esl correction tasks. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, Portland, Oregon, 6. Association for Computational Linguistics.
- Khaled Shaalan, Rana Aref, and Aly Fahmy. 2010. An approach for analyzing and correcting spelling errors for non-native Arabic learners. *Proceedings of Informatics and Systems (INFOS)*.
- L Venkata Subramaniam, Shourya Roy, Tanveer A Faruque, and Sumit Negi. 2009. A survey of types of text noise and techniques to handle noisy text. In *Proceedings of The Third Workshop on Analytics for Noisy Unstructured Text Data*, pages 115–122. ACM.
- Sebastian van Delden, David B. Bracewell, and Fernando Gomez. 2004. Supervised and unsupervised automatic spelling correction algorithms. In *Information Reuse and Integration, 2004. Proceedings of the 2004 IEEE International Conference on*, pages 530–535.
- Wajdi Zaghouani, Behrang Mohit, Nizar Habash, Osama Obeid, Nadi Tomeh, Alla Rozovskaya, Noura Farra, Sarah Alkuhlani, and Kemal Oflazer. 2014. Large scale Arabic error annotation: Guidelines and framework. In *Proceedings of the 9th edition of the Language Resources and Evaluation Conference*.

Improved Iterative Correction for Distant Spelling Errors

Sergey Gubanov

Irina Galinskaya

Alexey Baytin

Yandex

16 Leo Tolstoy St., Moscow, 119021 Russia

{esgv, galinskaya, baytin}@yandex-team.ru

Abstract

Noisy channel models, widely used in modern spellers, cope with typical misspellings, but do not work well with infrequent and difficult spelling errors. In this paper, we have improved the noisy channel approach by iterative stochastic search for the best correction. The proposed algorithm allowed us to avoid local minima problem and improve the F_1 measure by 6.6% on distant spelling errors.

1 Introduction

A speller is an essential part of any program associated with text input and processing — e-mail system, search engine, browser, form editor etc. To detect and correct spelling errors, the state of the art spelling correction systems use the noisy channel approach (Kernighan et al., 1990; Mays et al., 1991; Brill and Moore, 2000). Its models are usually trained on large corpora and provide high effectiveness in correction of typical errors (most of which consist of 1-2 wrong characters per word), but does not work well for complex (multi-character) and infrequent errors.

In this paper, we improved effectiveness of the noisy channel for the correction of complex errors. In most cases, these are cognitive errors in loan words (*folsvagen* → *volkswagen*), names of drugs (*vobemzin* → *wobenzym*), names of brands (*scatcher* → *skechers*), scientific terms (*heksagidron* → *hexahedron*) and last names (*Shwartzneger* → *Schwarzenegger*). In all these cases, the misspelled word contains many errors and the corresponding error model penalty cannot be compensated by the LM weight of its proper form. As a result, either the misspelled word itself, or the other (less complicated, more frequent) misspelling of the same word wins the likelihood race.

To compensate for this defect of the noisy channel, the iterative approach (Cucerzan and Brill, 2004) is typically used. The search for the best variant is repeated several times, what allows correcting rather complex errors, but does not completely solve the problem of falling into local minima. To overcome this issue we suggest to consider more correction hypotheses. For this purpose we used a method based on the simulated annealing algorithm. We experimentally demonstrate that the proposed method outperforms the baseline noisy channel and iterative spellers.

Many authors employ machine learning to build rankers that compensate for the drawbacks of the noisy channel model: (Whitelaw et al., 2009; Gao et al., 2010). These techniques can be combined with the proposed method by replacing posterior probability of single correction in our method with an estimate obtained via discriminative training method.

In our work, we focus on isolated word-error correction (Kukich, 1992), which, in a sense, is a harder task, than multi-word correction, because there is no context available for misspelled words. For experiments we used single-word queries to a commercial search engine.

2 Baseline speller

2.1 Noisy channel spelling correction

Noisy channel is a probabilistic model that defines posterior probability $P(q_0|q_1)$ of q_0 being the intended word, given the observed word q_1 ; for such model, the optimal decision rule μ is the following:

$$\begin{aligned} \mu(q_1) &= \arg \max_{q_0} P(q_0|q_1); \\ P(q_0|q_1) &\propto P_{\text{dist}}(q_0 \rightarrow q_1)P_{\text{LM}}(q_0), \end{aligned} \quad (1)$$

where P_{LM} is the source (language) model, and P_{dist} is the error model. Given $P(q_0|q_1)$ defined, to correct the word q_1 we could iterate through

all ever-observed words, and choose the one, that maximizes the posterior probability. However, the practical considerations demand that we do not rank the whole list of words, but instead choose between a limited number of hypotheses h_1, \dots, h_K :

1. Given q_1 , generate a set of hypotheses h_1, \dots, h_K , such that

$$\sum_{k=1}^K P(q_0 = h_k | q_1) \approx 1; \quad (2)$$

2. Choose the hypothesis h_k that maximizes $P(q_0 = h_k | q_1)$.

If hypotheses constitute a major part of the posterior probability mass, it is highly unlikely that the intended word is not among them.

2.2 Baseline speller setup

In baseline speller we use a substring-based error model $P_{\text{dist}}(q_0 \rightarrow q_1)$ described in (Brill and Moore, 2000), the error model training method and the hypotheses generator are similar to (Duan and Hsu, 2011).

For building language (P_{LM}) and error (P_{dist}) models, we use words collected from the 6-months query log of a commercial search engine.

Hypotheses generator is based on A* beam search in a trie of words, and yields K hypotheses h_k , for which the noisy channel scores $P_{\text{dist}}(h_k \rightarrow q_1)P_{\text{LM}}(h_k)$ are highest possible. Hypotheses generator has high K-best recall (see Section 4.2) — in 91.8% cases the correct hypothesis is found when $K = 30$, which confirms the assumption about covering almost all posterior probability mass (see Equation 2).

3 Improvements for noisy channel spelling correction

While choosing $\arg \max$ of the posterior probability is an optimal decision rule in theory, in practice it might not be optimal, due to limitations of the language and error modeling. For example, *vobemzin* is corrected to more frequent misspelling *vobenzin* (instead of correct form *wobenzym*) by the noisy channel, because $P_{\text{dist}}(\textit{vobemzin} \rightarrow \textit{wobenzym})$ is too low (see Table 1).

There have been attempts (Cucerzan and Brill, 2004) to apply other rules, which would overcome limitations of language and error models with compensating changes described further.

c	$-\log P_{\text{dist}}$	$-\log P_{\text{LM}}$	Σ
vobenzin	2.289	31.75	34.04
wobenzym	12.52	26.02	38.54

Table 1: Noisy-channel scores for two corrections of *vobemzin*

3.1 Iterative correction

Iterative spelling correction with E iterations uses standard noisy-channel to correct the query q repeatedly E times. It is motivated by the assumption, that we are more likely to successfully correct the query if we take several short steps instead of one big step (Cucerzan and Brill, 2004).

Iterative correction is hill climbing in the space of possible corrections: on each iteration we make a transition to the best point in the neighbourhood, i.e. to correction, that has maximal posterior probability $P(c|q)$. As any local search method, iterative correction is prone to local minima, stopping before reaching the correct word.

3.2 Stochastic iterative correction

A common method of avoiding local minima in optimization is the simulated annealing algorithm, key ideas from which can be adapted for spelling correction task. In this section we propose such an adaptation. Consider: we do not always transition deterministically to the next best correction, but instead transition randomly to a (potentially *any*) correction with transition probability being equal to the posterior $P(c_i | c_{i-1})$, where c_{i-1} is the correction we transition from, c_i is the correction we transition to, and $P(\cdot | \cdot)$ is defined by Equation 1. Iterative correction then turns into a *random walk*: we start at word $c_0 = q$ and stop after E random steps at some word c_E , which becomes our answer.

To turn random walk into deterministic spelling correction algorithm, we de-randomize it, using the following transformation. Described random walk defines, for each word w , a probability $P(c_E = w | q)$ of ending up in w after starting a walk from the initial query q . With that probability defined, our correction algorithm is the following: given query q , pick $c = \arg \max_{c_E} P(c_E | q)$ as a correction.

Probability of getting from $c_0 = q$ to some $c_E = c$ is a sum, over all possible paths, of probabilities of getting from q to c via specific path

$q = c_0 \rightarrow c_1 \rightarrow \dots \rightarrow c_{E-1} \rightarrow c_E = c$:

$$P(c_E|c_0) = \sum_{\substack{c_1 \in W \\ \dots \\ c_{E-1} \in W}} \prod_{i=1}^E P(c_i|c_{i-1}), \quad (3)$$

$$P(c_i|c_{i-1}) = \frac{P_{\text{dist}}(c_i \rightarrow c_{i-1})P_{\text{LM}}(c_i)}{P_{\text{observe}}(c_{i-1})}, \quad (4)$$

where W is the set of all possible words, and $P_{\text{observe}}(w)$ is the probability of observing w as a query in the noisy-channel model.

Example: if we start a random walk from *vobemzin* and make 3 steps, we most probably will end up in the correct form *wobenzym* with $P = 0.361$. A few of the most probable random walk paths are shown in Table 2. Note, that despite the fact that most probable path does not lead to the correct word, many other paths to *wobenzym* sum up to 0.361, which is greater than probability of any other word. Also note, that the method works only because multiple misspellings of the same word are presented in our model; for related research see (Choudhury et al., 2007).

$c_0 \rightarrow c_1 \rightarrow c_2 \rightarrow c_3$	P
vobemzin→vobenzin→vobenzin→vobenzin	0.074
vobemzin→vobenzim→wobenzym→ wobenzym	0.065
vobemzin→vobenzin→vobenzim→vobenzim	0.052
vobemzin→vobenzim→vobenzim→ wobenzym	0.034
vobemzin→wobenzym→wobenzym→ wobenzym	0.031
vobemzin→wobenzim→wobenzym→ wobenzym	0.028
vobemzin→wobenzyn→wobenzym→ wobenzym	0.022

Table 2: Most probable random walk paths starting from $c_0 = q = \textit{vobemzin}$ (the correct form is in bold).

Also note, that while Equation 3 uses noisy-channel posteriors, the method can use an arbitrary discriminative model, for example the one from (Gao et al., 2010), and benefit from a more accurate posterior estimate.

3.3 Additional heuristics

This section describes some common heuristic improvements, that, where possible, were applied both to the baseline methods and to the proposed algorithm.

Basic building block of every mentioned algorithm is one-step noisy-channel correction. Each basic correction proceeds as described in Section 2.1: a small number of hypotheses h_1, \dots, h_K is generated for the query q , hypotheses are scored,

and scores are recomputed into normalized posterior probabilities (see Equation 5). Posterior probabilities are then either used to pick the best correction (in baseline and simple iterative correction), or are accumulated to later compute the score defined by Equation 3.

$$\begin{aligned} \text{score}(h_i) &= P_{\text{dist}}(h_i \rightarrow q)^\lambda P_{\text{LM}}(h_i) \\ P(h_i|q) &= \text{score}(h_i) / \sum_{j=1}^K \text{score}(h_j) \end{aligned} \quad (5)$$

A standard log-linear weighing trick was applied to noisy-channel model components, see e.g. (Whitelaw et al., 2009). λ is the parameter that controls the trade-off between precision and recall (see Section 4.2) by emphasizing the importance of either the high frequency of the correction or its proximity to the query.

We have also found, that resulting posterior probabilities emphasize the best hypothesis too much: best hypothesis gets almost all probability mass and other hypotheses get none. To compensate for that, posteriors were *smoothed* by raising each probability to some power $\gamma < 1$ and re-normalizing them afterward:

$$P_{\text{smooth}}(h_i|q) = P(h_i|q)^\gamma / \sum_{j=1}^K P(h_j|q)^\gamma. \quad (6)$$

In a sense, γ is like temperature parameter in simulated annealing – it controls the entropy of the walk and the final probability distribution. Unlike in simulated annealing, we fix γ for all iterations of the algorithm.

Finally, if posterior probability of the best hypothesis was lower than threshold α , then the original query q was used as the spell-checker output. (Posterior is defined by Equation 6 for the baseline and simple iterative methods and by Equations 3 and 6 for the proposed method). Parameter α controls precision/recall trade-off (as well as λ mentioned above).

4 Experiments

4.1 Data

To evaluate the proposed algorithm we have collected two datasets. Both datasets were randomly sampled from single-word user queries from the 1-week query log of a commercial search engine. We annotated them with the help of professional analyst. The difference between datasets

is that one of them contained only queries with low search performance: for which the number of documents retrieved by the search engine was less than a fixed threshold (we will address it as the "hard" dataset), while the other dataset had no such restrictions (we will call it "common"). Dataset statistics are shown in Table 3.

Dataset	Queries	Misspelled	Avg. $-\log P_{\text{dist}}$
Common	2240	224 (10%)	5.98
Hard	2542	1484 (58%)	9.23

Table 3: Evaluation datasets.

Increased average error model score and error rate of "common" dataset compared to "hard" shows, that we have indeed managed to collect hard-to-correct queries in the "hard" dataset.

4.2 Experimental results

First of all, we evaluated the recall of hypotheses generator using *K-best recall* — the number of correct spelling corrections for misspelled queries among K hypotheses divided by the total number of misspelled queries in the test set. Resulting recall with $K = 30$ is 91.8% on "hard" and 98.6% on "common".

Next, three spelling correction methods were tested: noisy channel, iterative correction and our method (stochastic iterative correction).

For evaluation of spelling correction quality, we use the following metrics:

- *Precision*: The number of correct spelling corrections for misspelled words generated by the system divided by the total number of corrections generated by the system;
- *Recall*: The number of correct spelling corrections for misspelled words generated by the system divided by the total number of misspelled words in the test set;

For hypotheses generator, $K = 30$ was fixed: recall of 91.8% was considered big enough. Precision/recall tradeoff parameters λ and α (they are applicable to each method, including baseline) were iterated by the grid $(0.2, 0.25, 0.3, \dots, 1.5) \times (0, 0.025, 0.05, \dots, 1.0)$, and E (applicable to iterative and our method) and γ (just our method) were iterated by the grid $(2, 3, 4, 5, 7, 10) \times (0.1, 0.15, \dots, 1.0)$; for each set of parameters, precision and recall were measured on both datasets. Pareto frontiers for precision and recall are shown in Figures 1 and 2.

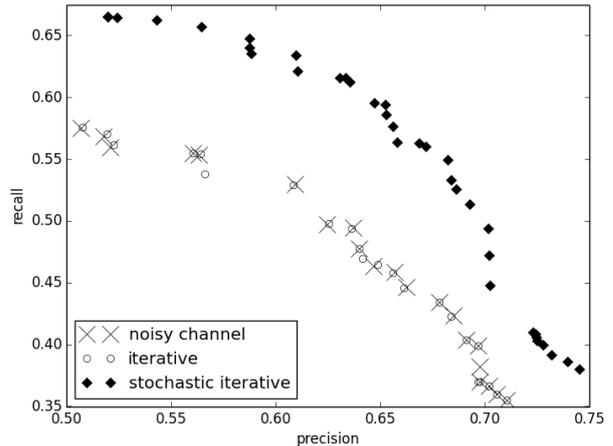


Figure 1: Precision/recall Pareto frontiers on "hard" dataset

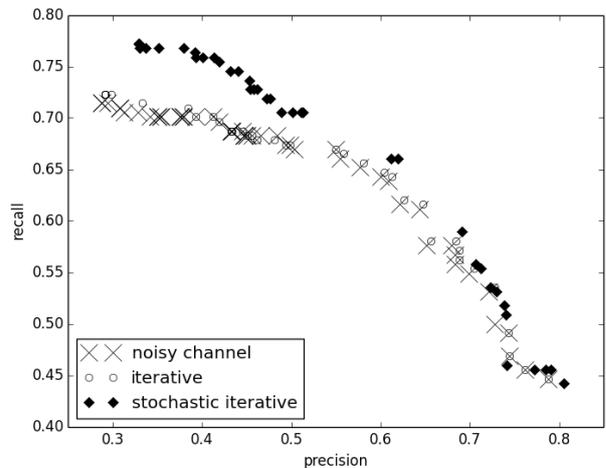


Figure 2: Precision/recall Pareto frontiers on "common" dataset

We were not able to reproduce superior performance of the iterative method over the noisy channel, reported by (Cucerzan and Brill, 2004). Supposedly, it is because the iterative method benefits primarily from the sequential application of split/join operations altering query decomposition into words; since we are considering only one-word queries, such decomposition does not matter.

On the "hard" dataset the performance of the noisy channel and the iterative methods is inferior to our proposed method, see Figure 1. We tested all three methods on the "common" dataset as well to evaluate if our handling of hard cases affects the performance of our approach on the common cases of spelling error. Our method performs well on the common cases as well, as Figure 2 shows. The performance comparison for the "common" dataset shows comparable performance for all considered methods.

Noisy channel and iterative methods' frontiers

are considerably inferior to the proposed method on "hard" dataset, which means that our method works better. The results on "common" dataset show, that the proposed method doesn't work worse than baseline.

Next, we optimized parameters for each method and each dataset separately to achieve the highest F_1 measure. Results are shown in Tables 4 and 5. We can see, that, given the proper tuning, our method can work better on any dataset (but it cannot achieve the best performance on both datasets at once). See Tables 4 and 5 for details.

Method	λ	α	γ	E	F_1
Noisy channel	0.6	0.1	-	-	55.8
Iterative	0.6	0.1	-	2	55.9
Stochastic iterative	0.9	0.2	0.35	3	62.5

Table 4: Best parameters and F_1 on "hard" dataset

Method	λ	α	γ	E	F_1
Noisy channel	0.75	0.225	-	-	62.06
Iterative	0.8	0.275	-	2	63.15
Stochastic iterative	1.2	0.4	0.35	3	63.9

Table 5: Best parameters and F_1 on "common" dataset

Next, each parameter was separately iterated (by a coarser grid); initial parameters for each method were taken from Table 4. Such iteration serves two purposes: to show the influence of parameters on algorithm performance, and to show differences between datasets: in such setup parameters are virtually tuned using "hard" dataset and evaluated using "common" dataset. Results are shown in Table 6.

The proposed method is able to successfully correct distant spelling errors with edit distance of 3 characters (see Table 7).

However, if our method is applied to shorter and more frequent queries (as opposed to "hard" dataset), it tends to suggest frequent words as false-positive corrections (for example, *grid* is corrected to *creed* – Assassin's Creed is popular video game). As can be seen in Table 5, in order to fix that, algorithm parameters need to be tuned more towards precision.

5 Conclusion and future work

In this paper we introduced the stochastic iterative correction method for spell check corrections. Our experimental evaluation showed that the proposed method improved the performance of popu-

	F_1 , common			F_1 , hard		
	N.ch.	It.	Our	N.ch.	It.	Our
$\lambda = 0.5$	45.3	45.9	37.5	54.9	54.8	50.0
0.6	49.9	50.5	41.5	55.8	55.9	56.6
0.7	50.4	50.4	44.1	54.5	55.1	59.6
0.8	52.7	52.7	46.0	52.6	53.0	61.5
0.9	53.5	53.5	49.3	50.3	50.6	62.5
1.0	55.4	55.0	50.9	47.0	47.3	61.8
1.1	53.7	53.4	52.7	44.3	44.6	60.8
1.2	52.5	52.5	53.7	41.9	42.3	58.8
1.3	52.2	52.6	54.6	39.5	39.9	56.6
1.4	51.4	51.8	55.0	36.8	37.3	53.6
$\alpha = 0$	41.0	41.5	33.0	52.9	53.1	58.3
0.1	49.9	50.6	35.6	55.8	55.9	59.7
0.15	59.4	59.8	43.2	55.8	55.6	61.6
0.2	60.8	61.3	49.4	51.0	51.0	62.5
0.25	54.0	54.0	54.9	46.3	46.3	61.1
0.3	46.3	46.3	57.3	39.2	39.2	58.4
0.4	25.8	25.8	53.9	22.3	22.3	50.3
$E = 2$		50.6	53.6		55.9	60.4
3		50.6	49.4		55.9	62.5
4		50.6	46.4		55.9	62.1
5		50.6	46.7		55.9	60.1
$\gamma = 0.1$			10.1			6.0
0.2			49.4			51.5
0.3			51.4			61.4
0.35			49.4			62.5
0.4			47.5			62.0
0.45			45.8			60.8
0.5			45.2			60.3

Table 6: Per-coordinate iteration of parameters from Table 4; per-method maximum is shown in italic, per-dataset in bold

Query	Noisy channel	Proposed method
akwamarin	akvamarin	aquamarine
maccartni	maccartni	mccartney
ariflaim	ariflaim	oriflame
epika	epica	replica
grid	grid	creed

Table 7: Correction examples for the noisy channel and the proposed method.

lar spelling correction approach – the noisy channel model – in the correction of difficult spelling errors. We showed how to eliminate the local minima issue of simulated annealing and proposed a technique to make our algorithm deterministic.

The experiments conducted on the specialized datasets have shown that our method significantly improves the performance of the correction of hard spelling errors (by 6.6% F_1) while maintaining good performance on common spelling errors.

In continuation of the work we are considering to expand the method to correct errors in multi-word queries, extend the method to work with discriminative models, and use a query performance prediction method, which tells for a query whether our algorithm needs to be applied.

References

- Eric Brill and Robert C Moore. 2000. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 286–293. Association for Computational Linguistics.
- Monojit Choudhury, Markose Thomas, Animesh Mukherjee, Anupam Basu, and Niloy Ganguly. 2007. How difficult is it to develop a perfect spell-checker? a cross-linguistic analysis through complex network approach. In *Proceedings of the second workshop on TextGraphs: Graph-based algorithms for natural language processing*, pages 81–88.
- Silviu Cucerzan and Eric Brill. 2004. Spelling correction as an iterative process that exploits the collective knowledge of web users. In *EMNLP*, volume 4, pages 293–300.
- Huizhong Duan and Bo-June Paul Hsu. 2011. Online spelling correction for query completion. In *Proceedings of the 20th international conference on World wide web*, pages 117–126. ACM.
- Jianfeng Gao, Xiaolong Li, Daniel Micol, Chris Quirk, and Xu Sun. 2010. A large scale ranker-based system for search query spelling correction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 358–366. Association for Computational Linguistics.
- Mark D Kernighan, Kenneth W Church, and William A Gale. 1990. A spelling correction program based on a noisy channel model. In *Proceedings of the 13th conference on Computational linguistics-Volume 2*, pages 205–210. Association for Computational Linguistics.
- Karen Kukich. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys (CSUR)*, 24(4):377–439.
- Eric Mays, Fred J Damerau, and Robert L Mercer. 1991. Context based spelling correction. *Information Processing & Management*, 27(5):517–522.
- Casey Whitelaw, Ben Hutchinson, Grace Y Chung, and Gerard Ellis. 2009. Using the web for language independent spellchecking and autocorrection. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 890–899. Association for Computational Linguistics.

Predicting Grammaticality on an Ordinal Scale

Michael Heilman Aoife Cahill Nitin Madnani Melissa Lopez Matthew Mulholland

Educational Testing Service

Princeton, NJ, USA

{mheilman, acahill, nmadnani, mlopez002, mmulholland}@ets.org

Joel Tetreault

Yahoo! Research

New York, NY, USA

tetreaul@yahoo-inc.com

Abstract

Automated methods for identifying whether sentences are grammatical have various potential applications (e.g., machine translation, automated essay scoring, computer-assisted language learning). In this work, we construct a statistical model of grammaticality using various linguistic features (e.g., misspelling counts, parser outputs, n -gram language model scores). We also present a new publicly available dataset of learner sentences judged for grammaticality on an ordinal scale. In evaluations, we compare our system to the one from Post (2011) and find that our approach yields state-of-the-art performance.

1 Introduction

In this paper, we develop a system for the task of predicting the grammaticality of sentences, and present a dataset of learner sentences rated for grammaticality. Such a system could be used, for example, to check or to rank outputs from systems for text summarization, natural language generation, or machine translation. It could also be used in educational applications such as essay scoring.

Much of the previous research on predicting grammaticality has focused on identifying (and possibly correcting) specific types of grammatical errors that are typically made by English language learners, such as prepositions (Tetreault and Chodorow, 2008), articles (Han et al., 2006), and collocations (Dahlmeier and Ng, 2011). While some applications (e.g., grammar checking) rely on such fine-grained predictions, others might be better addressed by sentence-level grammaticality judgments (e.g., machine translation evaluation).

Regarding sentence-level grammaticality, there has been much work on rating the grammatical-

ity of machine translation outputs (Gamon et al., 2005; Parton et al., 2011), such as the MT Quality Estimation Shared Tasks (Bojar et al., 2013, §6), but relatively little on evaluating the grammaticality of naturally occurring text. Also, most other research on evaluating grammaticality involves *artificial* tasks or datasets (Sun et al., 2007; Lee et al., 2007; Wong and Dras, 2010; Post, 2011).

Here, we make the following contributions.

- We develop a state-of-the-art approach for predicting the grammaticality of sentences on an ordinal scale, adapting various techniques from the previous work described above.
- We create a dataset of grammatical and ungrammatical sentences written by English language learners, labeled on an ordinal scale for grammaticality. With this unique data set, which we will release to the research community, it is now possible to conduct realistic evaluations for predicting sentence-level grammaticality.

2 Dataset Description

We created a dataset consisting of 3,129 sentences randomly selected from essays written by non-native speakers of English as part of a test of English language proficiency. We oversampled lower-scoring essays to increase the chances of finding ungrammatical sentences. Two of the authors of this paper, both native speakers of English with linguistic training, annotated the data. We refer to these annotators as expert judges. When making judgments of the sentences, they saw the previous sentence from the same essay as context. These two authors were not directly involved in development of the system in §3.

Each sentence was annotated on a scale from 1 to 4 as described below, with 4 being the most

grammatical. We use an ordinal rather than binary scale, following previous work such as that of Clark et al. (2013) and Crocker and Keller (2005) who argue that the distinction between grammatical and ungrammatical is not simply binary. Also, for practical applications, we believe that it is useful to distinguish sentences with minor errors from those with major errors that may disrupt communication. Our annotation scheme was influenced by a translation rating scheme by Coughlin (2003).

Every sentence judged on the 1–4 scale must be a clause. There is an extra category (“Other”) for sentences that do not fit this criterion. We exclude instances of “Other” in our experiments (see §4).

4. Perfect The sentence is native-sounding. It has no grammatical errors, but may contain very minor typographical and/or collocation errors, as in Example (1).

- (1) For instance, i stayed in a dorm when i went to collge.

3. Comprehensible The sentence may contain one or more minor grammatical errors, including subject-verb agreement, determiner, and minor preposition errors that do not make the meaning unclear, as in Example (2).

- (2) We know during Spring Festival, Chinese family will have a abundand family banquet with family memebbers.

“Chinese family”, which could be corrected to “Chinese families”, “each Chinese family”, etc., would be an example of a minor grammatical error involving determiners.

2. Somewhat Comprehensible The sentence may contain one or more serious grammatical errors, including missing subject, verb, object, etc., verb tense errors, and serious preposition errors. Due to these errors, the sentence may have multiple plausible interpretations, as in Example (3).

- (3) I can gain the transportations such as buses and trains.

1. Incomprehensible The sentence contains so many errors that it would be difficult to correct, as in Example (4).

- (4) Or you want to say he is only a little boy do not everything clearly?

The phrase “do not everything” makes the sentence practically incomprehensible since the subject of “do” is not clear.

O. Other/Incomplete This sentence is incomplete. These sentences, such as Example (5), appear in our corpus due to the nature of timed tests.

- (5) The police officer handed the

This sentence is cut off and does not at least include one clause.

We measured interannotator agreement on a subset of 442 sentences that were independently annotated by both expert annotators. Exact agreement was 71.3%, unweighted $\kappa = 0.574$, and Pearson’s $r = 0.759$.¹ For our experiments, one expert annotator was arbitrarily selected, and for the doubly-annotated sentences, only the judgments from that annotator were retained.

The labels from the expert annotators are distributed as follows: 72 sentences are labeled 1; 538 are 2; 1,431 are 3; 978 are 4; and 110 are “O”.

We also gathered 5 additional judgments using Crowdfunder.² For this, we excluded the “Other” category and any sentences that had been marked as such by the expert annotators. We used 100 (3.2%) of the judged sentences as “gold” data in Crowdfunder to block contributors who were not following the annotation guidelines. For those sentences, only disagreements within 1 point of the expert annotator judgment were accepted. In preliminary experiments, averaging the six judgments (1 expert, 5 crowdsourced) for each item led to higher human-machine agreement. For all experiments reported later, we used this average of six judgments as our gold standard.

For our experiments (§4), we randomly split the data into training (50%), development (25%), and testing (25%) sets. We also excluded all instances labeled “Other”. These are relatively uncommon and less interesting to this study. Also, we believe that simpler, heuristic approaches could be used to identify such sentences.

We use “GUG” (“Grammatical” versus “Un-Grammatical”) to refer to this dataset. The dataset is available for research at <https://github.com/EducationalTestingService/gug-data>.

¹The reported agreement values assume that “Other” maps to 0. For the sentences where both labels were in the 1–4 range ($n = 424$), Pearson’s $r = 0.767$.

²<http://www.crowdfunder.com>

3 System Description

This section describes the statistical model (§3.1) and features (§3.2) used by our system.

3.1 Statistical Model

We use ℓ_2 -regularized linear regression (i.e., ridge regression) to learn a model of sentence grammaticality from a variety of linguistic features.³⁴

To tune the ℓ_2 -regularization hyperparameter α , the system performs 5-fold cross-validation on the data used for training. The system evaluates $\alpha \in 10^{\{-4, \dots, 4\}}$ and selects the one that achieves the highest cross-validation correlation r .

3.2 Features

Next, we describe the four types of features.

3.2.1 Spelling Features

Given a sentence with n word tokens, the model filters out tokens containing nonalphabetic characters and then computes the number of misspelled words n_{miss} (later referred to as `num_misspelled`), the proportion of misspelled words $\frac{n_{miss}}{n}$, and $\log(n_{miss} + 1)$ as features. To identify misspellings, we use a freely available spelling dictionary for U.S. English.⁵

3.2.2 n -gram Count and Language Model Features

Given each sentence, the model obtains the counts of n -grams ($n = 1 \dots 3$) from English Gigaword and computes the following features:⁶

$$\bullet \sum_{s \in S_n} \frac{\log(\text{count}(s) + 1)}{\|S_n\|}$$

³We use ridge regression from the `scikit-learn` toolkit (Pedregosa et al., 2011) v0.23.1 and the SciKit-Learn Laboratory (<http://github.com/EducationalTestingService/skll>).

⁴Regression models typically produce conservative predictions with lower variance than the original training data. So that predictions better match the distribution of labels in the training data, the system rescales its predictions. It saves the mean and standard deviation of the training data gold standard (M_{gold} and SD_{gold} , respectively) and of its own predictions on the training data (M_{pred} and SD_{pred} , respectively). During cross-validation, this is done for each fold. From an initial prediction \hat{y} , it produces the final prediction: $\hat{y}' = \frac{\hat{y} - M_{pred}}{SD_{pred}} * SD_{gold} + M_{gold}$. This transformation does not affect Pearson's r correlations or rankings, but it would affect binarized predictions.

⁵<http://pythonhosted.org/pyenchant/>

⁶We use the New York Times (`nyt`), the Los Angeles Times-Washington Post (`ltw`), and the Washington Post-Bloomberg News (`wpb`) sections from the fifth edition of English Gigaword (LDC2011T07).

- $\max_{s \in S_n} \log(\text{count}(s) + 1)$
- $\min_{s \in S_n} \log(\text{count}(s) + 1)$

where S_n represents the n -grams of order n from the given sentence. The model computes the following features from a 5-gram language model trained on the same three sections of English Gigaword using the SRILM toolkit (Stolcke, 2002):

- the average log-probability of the given sentence (referred to as `gigaword_avglogprob` later)
- the number of out-of-vocabulary words in the sentence

Finally, the system computes the average log-probability and number of out-of-vocabulary words from a language model trained on a collection of essays written by non-native English speakers⁷ (“non-native LM”).

3.2.3 Precision Grammar Features

Following Wagner et al. (2007) and Wagner et al. (2009), we use features extracted from precision grammar parsers. These grammars have been hand-crafted and designed to only provide complete syntactic analyses for grammatically correct sentences. This is in contrast to treebank-trained grammars, which will generally provide *some* analysis regardless of grammaticality. Here, we use (1) the Link Grammar Parser⁸ and (2) the HPSG English Resource Grammar (Copestake and Flickinger, 2000) and PET parser.⁹

We use a binary feature, `complete_link`, from the Link grammar that indicates whether at least one complete linkage can be found for a sentence. We also extract several features from the HPSG analyses.¹⁰ They mostly reflect information about unification success or failure and the associated costs. In each instance, we use the logarithm of one plus the frequency.

⁷This did not overlap with the data described in §2 and was a subset of the data released by Blanchard et al. (2013).

⁸<http://www.link.cs.cmu.edu/link/>

⁹<http://moin.delph-in.net/PetTop>

¹⁰The complete list of relevant statistics used as features is: `trees`, `unify_cost_succ`, `unify_cost_fail`, `unifications_succ`, `unifications_fail`, `subsumptions_succ`, `subsumptions_fail`, `words`, `words_pruned`, `aedges`, `pedges`, `upedges`, `raedges`, `rpedges`, `medges`. During development, we observed that some of these features vary for some inputs, probably due to parsing search timeouts. On 10 preliminary runs with the development set, this variance had minimal effects on correlations with human judgments (less than 0.00001 in terms of r).

	r
our system	0.668
– non-native LM (§3.2.2)	0.665
– HPSG parse (§3.2.3)	0.664
– PCFG parse (§3.2.4)	0.662
– spelling (§3.2.1)	0.643
– gigaword LM (§3.2.2)	0.638
– link parse (§3.2.3)	0.632
– gigaword count (§3.2.2)	0.630

Table 1: Pearson’s r on the development set, for our full system and variations excluding each feature type. “– X ” indicates the full model without the “ X ” features.

3.2.4 PCFG Parsing Features

We find phrase structure trees and basic dependencies with the Stanford Parser’s English PCFG model (Klein and Manning, 2003; de Marneffe et al., 2006).¹¹ We then compute the following:

- the parse score as provided by the Stanford PCFG Parser, normalized for sentence length, later referred to as `parse_prob`
- a binary feature that captures whether the top node of the tree is sentential or not (i.e. the assumption is that if the top node is non-sentential, then the sentence is a fragment)
- features binning the number of `dep` relations returned by the dependency conversion. These `dep` relations are underspecified for function and indicate that the parser was unable to find a standard relation such as `subj`, possibly indicating a grammatical error.

4 Experiments

Next, we present evaluations on the GUG dataset.

4.1 Feature Ablation

We conducted a feature ablation study to identify the contributions of the different types of features described in §3.2. We compared the performance of the full model with all of the features to models with all but one type of feature. For this experiment, all models were estimated from the training set and evaluated on the development set. We report performance in terms of Pearson’s r between the averaged 1–4 human labels and unrounded system predictions.

The results are shown in Table 1. From these results, the most useful features appear to be the

n -gram frequencies from Gigaword and whether the link parser can fully parse the sentence.

4.2 Test Set Results

In this section, we present results on the held-out test set for the full model and various baselines, summarized in Table 2. For test set evaluations, we trained on the combination of the training and development sets (§2), to maximize the amount of training data for the final experiments.

We also trained and evaluated on binarized versions of the ordinal GUG labels: a sentence was labeled 1 if the average judgment was at least 3.5 (i.e., would round to 4), and 0 otherwise. Evaluating on a binary scale allows us to measure how well the system distinguishes grammatical sentences from ungrammatical ones. For some applications, this two-way distinction may be more relevant than the more fine-grained 1–4 scale. To train our system on binarized data, we replaced the ℓ_2 -regularized linear regression model with an ℓ_2 -regularized logistic regression and used Kendall’s τ rank correlation between the predicted probabilities of the positive class and the binary gold standard labels as the grid search metric (§3.1) instead of Pearson’s r .

For the ordinal task, we report Pearson’s r between the averaged human judgments and each system. For the binary task, we report percentage accuracy. Since the predictions from the binary and ordinal systems are on different scales, we include the nonparametric statistic Kendall’s τ as a secondary evaluation metric for both tasks.

We also evaluated the binary system for the ordinal task by computing correlations between its estimated probabilities and the averaged human scores, and we evaluated the ordinal system for the binary task by binarizing its predictions.¹²

We compare our work to a modified version of the publicly available¹³ system from Post (2011), which performed very well on an artificial dataset. To our knowledge, it is the only publicly available system for grammaticality prediction. It is very

¹¹We use the Nov. 12, 2013 version of the Stanford Parser.

¹²We selected a threshold for binarization from a grid of 1001 points from 1 to 4 that maximized the accuracy of binarized predictions from a model trained on the training set and evaluated on the binarized development set. For evaluating the three single-feature baselines discussed below, we used the same approach except with grid ranging from the minimum development set feature value to the maximum plus 0.1% of the range.

¹³The Post (2011) system is available at <https://github.com/mjpost/post2011judging>.

	Ordinal Task			Binary Task		
	r	$Sig.r$	τ	% Acc.	$Sig.\%Acc.$	τ
our system	0.644		0.479	79.3		0.419
our system _{logistic}	0.616	*	0.484	80.7		0.428
Post	0.321	*	0.225	75.5	*	0.195
Post _{logistic}	0.259	*	0.181	74.4	*	0.181
complete_link	0.386	*	0.335	74.8	*	0.302
gigaword_avglogprob	0.414	*	0.290	76.7	*	0.280
num_misspelled	-0.462	*	-0.370	74.8	*	-0.335

Table 2: Human-machine agreement statistics for our system, the system from Post (2011), and simple baselines, computed from the averages of human ratings in the testing set (§2). “*” in a Sig. column indicates a statistically significant difference from “our system” ($p < .05$, see text for details). A majority baseline for the binary task achieves 74.8% accuracy. The best results for each metric are in bold.

different from our system since it relies on partial tree-substitution grammar derivations as features. We use the feature computation components of that system but replace its statistical model. The system was designed for use with a dataset consisting of 50% grammatical and 50% ungrammatical sentences, rather than data with ordinal or continuous labels. Additionally, its classifier implementation does not output scores or probabilities. Therefore, we used the same learning algorithms as for our system (i.e., ridge regression for the ordinal task and logistic regression for the binary task).¹⁴

To create further baselines for comparison, we selected the following features that represent ways one might approximate grammaticality if a comprehensive model was unavailable: whether the link parser can fully parse the sentence (`complete_link`), the Gigaword language model score (`gigaword_avglogprob`), and the number of misspelled tokens (`num_misspelled`). Note that we expect the number of misspelled tokens to be negatively correlated with grammaticality. We flipped the sign of the misspelling feature when computing accuracy for the binary task.

To identify whether the differences in performance for the ordinal task between our system and each of the baselines are statistically significant, we used the BC_a Bootstrap (Efron and Tibshirani, 1993) with 10,000 replications to compute 95% confidence intervals for the absolute value of r for our system minus the absolute value of r for each of the alternative methods. For the binary task, we

¹⁴In preliminary experiments, we observed little difference in performance between logistic regression and the original support vector classifier used by the system from Post (2011).

used the sign test to test for significant differences in accuracy. The results are in Table 2.

5 Discussion and Conclusions

In this paper, we developed a system for predicting grammaticality on an ordinal scale and created a labeled dataset that we have released publicly (§2) to enable more realistic evaluations in future research. Our system outperformed an existing state-of-the-art system (Post, 2011) in evaluations on binary and ordinal scales. This is the most realistic evaluation of methods for predicting sentence-level grammaticality to date.

Surprisingly, the system from Post (2011) performed quite poorly on the GUG dataset. We speculate that this is due to the fact that the Post system relies heavily on features extracted from automatic syntactic parses. While Post found that such a system can effectively distinguish grammatical news text sentences from sentences generated by a language model, measuring the grammaticality of real sentences from language learners seems to require a wider variety of features, including n -gram counts, language model scores, etc. Of course, our findings do not indicate that syntactic features such as those from Post (2011) are without value. In future work, it may be possible to improve grammaticality measurement by integrating such features into a larger system.

Acknowledgements

We thank Beata Beigman Klebanov, Yoko Futagi, Su-Youn Yoon, and the anonymous reviewers for their helpful comments. We also thank Jennifer Foster for discussions about this work and Matt Post for making his system publicly available.

References

- Daniel Blanchard, Joel Tetreault, Derrick Higgins, Aoife Cahill, and Martin Chodorow. 2013. TOEFL11: A Corpus of Non-Native English. Technical report, Educational Testing Service.
- 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, August. Association for Computational Linguistics.
- Alexander Clark, Gianluca Giorgolo, and Shalom Lapin. 2013. Towards a statistical model of grammaticality. In *Proceedings of the 35th Annual Conference of the Cognitive Science Society*, pages 2064–2069.
- Ann Copestake and Dan Flickinger. 2000. An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC 2000)*, Athens, Greece.
- Deborah Coughlin. 2003. Correlating automated and human assessments of machine translation quality. In *Proceedings of MT Summit IX*, pages 63–70.
- Matthew W. Crocker and Frank Keller. 2005. Probabilistic grammars as models of gradience in language processing. In *Gradience in Grammar: Generative Perspectives*. University Press.
- Daniel Dahlmeier and Hwee Tou Ng. 2011. Correcting Semantic Collocation Errors with L1-induced Paraphrases. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *LREC 2006*, pages 449–454.
- B. Efron and R. Tibshirani. 1993. *An Introduction to the Bootstrap*. Chapman and Hall/CRC, Boca Raton, FL.
- Michael Gamon, Anthony Aue, and Martine Smets. 2005. Sentence-level MT evaluation without reference translations: Beyond language modeling. In *Proceedings of EAMT*, pages 103–111. Springer-Verlag.
- Na-Rae Han, Martin Chodorow, and Claudia Leacock. 2006. Detecting errors in English article usage by non-native speakers. *Natural Language Engineering*, 12(2):115–129.
- Dan Klein and Christopher D. Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan, July. Association for Computational Linguistics.
- John Lee, Ming Zhou, and Xiaohua Liu. 2007. Detection of Non-Native Sentences Using Machine-Translated Training Data. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 93–96, Rochester, New York, April. Association for Computational Linguistics.
- Kristen Parton, Joel Tetreault, Nitin Madnani, and Martin Chodorow. 2011. E-rating machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 108–115, Edinburgh, Scotland, July. Association for Computational Linguistics.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.
- Matt Post. 2011. Judging Grammaticality with Tree Substitution Grammar Derivations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 217–222, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Andreas Stolcke. 2002. SRILM - An Extensible Language Modeling Toolkit. In *7th International Conference on Spoken Language Processing*.
- Guihua Sun, Xiaohua Liu, Gao Cong, Ming Zhou, Zhongyang Xiong, John Lee, and Chin-Yew Lin. 2007. Detecting Erroneous Sentences using Automatically Mined Sequential Patterns. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 81–88, Prague, Czech Republic, June. Association for Computational Linguistics.
- Joel R. Tetreault and Martin Chodorow. 2008. The Ups and Downs of Preposition Error Detection in ESL Writing. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 865–872, Manchester, UK, August. Coling 2008 Organizing Committee.
- Joachim Wagner, Jennifer Foster, and Josef van Genabith. 2007. A Comparative Evaluation of Deep and Shallow Approaches to the Automatic Detection of Common Grammatical Errors. In *Proceedings of the 2007 Joint Conference on Empirical*

Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pages 112–121, Prague, Czech Republic, June. Association for Computational Linguistics.

Joachim Wagner, Jennifer Foster, and Josef van Genabith. 2009. Judging grammaticality: Experiments in sentence classification. *CALICO Journal*, 26(3):474–490.

Sze-Meng Jojo Wong and Mark Dras. 2010. Parser Features for Sentence Grammaticality Classification. In *Proceedings of the Australasian Language Technology Association Workshop 2010*, pages 67–75, Melbourne, Australia, December.

I'm a Belieber: Social Roles via Self-identification and Conceptual Attributes

Charley Beller, Rebecca Knowles, Craig Harman
Shane Bergsma[†], Margaret Mitchell[‡], Benjamin Van Durme

Human Language Technology Center of Excellence
Johns Hopkins University, Baltimore, MD USA

[†]University of Saskatchewan, Saskatoon, Saskatchewan Canada

[‡]Microsoft Research, Redmond, Washington USA

charleybeller@jhu.edu, rknowles@jhu.edu, craig@craigharman.net,
shane.a.bergsma@gmail.com, memitc@microsoft.com, vandurme@cs.jhu.edu

Abstract

Motivated by work predicting coarse-grained author categories in social media, such as gender or political preference, we explore whether Twitter contains information to support the prediction of *fine-grained* categories, or *social roles*. We find that the simple self-identification pattern “*I am a ___*” supports significantly richer classification than previously explored, successfully retrieving a variety of fine-grained roles. For a given role (e.g., **writer**), we can further identify characteristic *attributes* using a simple possessive construction (e.g., *writer's ___*). Tweets that incorporate the attribute terms in first person possessives (*my ___*) are confirmed to be an indicator that the author holds the associated social role.

1 Introduction

With the rise of social media, researchers have sought to induce models for predicting latent *author attributes* such as gender, age, and political preferences (Garera and Yarowsky, 2009; Rao et al., 2010; Burger et al., 2011; Van Durme, 2012b; Zamal et al., 2012). Such models are clearly in line with the goals of both computational advertising (Wortman, 2008) and the growing area of computational social science (Conover et al., 2011; Nguyen et al., 2011; Paul and Dredze, 2011; Pennacchiotti and Popescu, 2011; Mohammad et al., 2013) where big data and computation supplement methods based on, e.g., direct human surveys. For example, Eisenstein et al. (2010) demonstrated a model that predicted where an author was located in order to analyze regional distinctions in communication. While some users explicitly share their GPS coordinates through their

Twitter clients, having a larger collection of automatically identified users within a region was preferable even though the predictions for any given user were uncertain.

We show that media such as Twitter can support classification that is more fine-grained than gender or general location. Predicting *social roles* such as **doctor**, **teacher**, **vegetarian**, **christian**, may open the door to large-scale passive surveys of public discourse that dwarf what has been previously available to social scientists. For example, work on tracking the spread of flu infections across Twitter (Lamb et al., 2013) might be enhanced with a factor based on aggregate predictions of author occupation.

We present two studies showing that first-person social content (tweets) contains intuitive signals for such fine-grained roles. We argue that non-trivial classifiers may be constructed based purely on leveraging simple linguistic patterns. These baselines suggest a wide range of author categories to be explored further in future work.

Study 1 In the first study, we seek to determine whether such a signal exists in *self-identification*: we rely on variants of a single pattern, “*I am a ___*”, to bootstrap data for training balanced-class binary classifiers using unigrams observed in tweet content. As compared to prior research that required actively polling users for ground truth in order to construct predictive models for demographic information (Kosinski et al., 2013), we demonstrate that some users specify such properties publicly through direct natural language.

Many of the resultant models show intuitive strongly-weighted features, such as a **writer** being likely to tweet about a *story*, or an **athlete** discussing a *game*. This demonstrates self-identification as a viable signal in building predictive models of social roles.

Role	Tweet
artist	I'm an Artist..... the last of a dying breed
belieber	@justinbieber I will support you in everything you do because I am a belieber please follow me I love you 30
vegetarian	So glad I'm a vegetarian.

Table 1: Examples of self-identifying tweets.

#	Role	#	Role	#	Role
29,924	little	5,694	man	564	champion
21,822	big	559	teacher
18,957	good	4,007	belieber	556	writer
13,069	huge	3,997	celebrity	556	awful
13,020	bit	3,737	virgin
12,816	fan	3,682	pretty	100	cashier
10,832	bad	100	bro
10,604	girl	2,915	woman
9,981	very	2,851	beast	10	linguist
...

Table 2: Number of self-identifying users per “role”. While rich in interesting labels, cases such as *very* highlight the purposeful simplicity of the current approach.

Study 2 In the second study we exploit a complementary signal based on characteristic *conceptual attributes* of a social role, or concept class (Schubert, 2002; Almuhabeb and Poesio, 2004; Paşca and Van Durme, 2008). We identify typical attributes of a given social role by collecting terms in the Google n-gram corpus that occur frequently in a possessive construction with that role. For example, with the role **doctor** we extract terms matching the simple pattern “*doctor’s* ___”.

2 Self-identification

All role-representative users were drawn from the free public 1% sample of the Twitter Firehose, over the period 2011-2013, from the subset that selected English as their native language (85,387,204 unique users). To identify users of a particular role, we performed a case-agnostic search of variants of a single pattern: *I am a(n)* __, and *I’m a(n)* __, where all single tokens filling the slot were taken as evidence of the author self-reporting for the given “role”. Example tweets can be seen in Table 1, examples of frequency per role in Table 2. This resulted in 63,858 unique roles identified, of which 44,260 appeared only once.¹

We manually selected a set of roles for further exploration, aiming for a diverse sample across: occupation (e.g., **doctor**, **teacher**), family (**mother**), disposition (**pessimist**), religion (**chris-**

¹Future work should consider identifying multi-word role labels (e.g., *Doctor Who fan*, or *dog walker*).

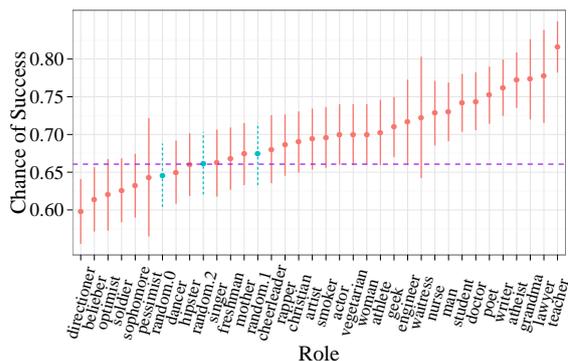


Figure 1: Success rate for querying a user. Random.0,1,2 are background draws from the population, with the mean of those three samples drawn horizontally. Tails capture 95% confidence intervals.

tian), and “followers” (**belieber**, **directioner**).² We filtered users via language ID (Bergsma et al., 2012) to better ensure English content.³

For each selected role, we randomly sampled up to 500 unique self-reporting users and then queried Twitter for up to 200 of their recent publicly posted tweets.⁴ These tweets served as representative content for that role, with any tweet matching the self-reporting patterns filtered. Three sets of background populations were extracted based on randomly sampling users that self-reported English (post-filtered via LID).

Twitter users are empowered to at any time delete, rename or make private their accounts. Any given user taken to be representative based on a previously posted tweet may no longer be available to query on. As a hint of the sort of user studies one might explore given access to social role prediction, we see in Figure 1 a correlation between self-reported role and the chance of an account still being publicly visible, with roles such as **belieber** and **directioner** on the one hand, and **doctor** and **teacher** on the other.

The authors examined the self-identifying tweet of 20 random users per role. The accuracy of the self-identification pattern varied across roles and is attributable to various factors including quotes, e.g. *@StarTrek Jim, I’m a DOCTOR not a download!*. While these samples are small (and thus estimates of quality come with wide variance), it

²Those that follow the music/life of the singer Justin Bieber and the band One Direction, respectively.

³This removes users that selected English as their primary language, used a self-identification phrase, e.g. *I am a belieber*, but otherwise tended to communicate in non-English.

⁴Roughly half of the classes had less than 500 self-reporting users in total, in those cases we used all matches.

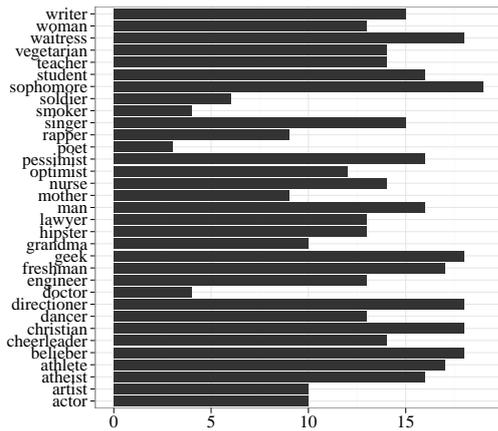


Figure 2: Valid self-identifying tweets from sample of 20.

is noteworthy that a non-trivial number for each were judged as actually self-identifying.

Indicative Language Most work in user classification relies on featurizing language use, most simply through binary indicators recording whether a user did or did not use a particular word in a history of n tweets. To explore whether language provides signal for future work in fine-grain social role prediction, we constructed a set of experiments, one per role, where training and test sets were balanced between users from a random background sample and self-reported users. Baseline accuracy in these experiments was thus 50%.

Each training set had a target of 600 users (300 background, 300 self-identified); for those roles with less than 300 users self-identifying, all users were used, with an equal number background. We used the `Jerboa` (Van Durme, 2012a) platform to convert data to binary feature vectors over a unigram vocabulary filtered such that the minimum frequency was 5 (across unique users). Training and testing was done with a log-linear model via `LibLinear` (Fan et al., 2008). We used the positively annotated data to form test sets, balanced with data from the background set. Each test set had a theoretical maximum size of 40, but for several classes it was in the single digits (see Figure 2). Despite the varied noisiness of our simple pattern-bootstrapped training data, and the small size of our annotated test set, we see in Figure 3 that we are able to successfully achieve statistically significant predictions of social role for the majority of our selected examples.

Table 3 highlights examples of language indicative of role, as determined by the most positively weighted unigrams in the classification experi-

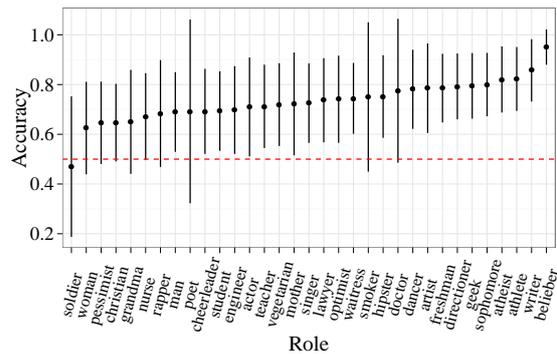


Figure 3: Accuracy in classifying social roles.

Role	Feature (Rank)
artist	morning, summer, life, most, amp, studio
atheist	fuck, fucking, shit, makes, dead, ..., religion ₁₉
athlete	lol, game, probably, life, into, ..., team ₉
belieber	justin, justinbeiber, believe, beliebers, bieber
cheerleader	cheer, best, excited, hate, mom, ..., prom ₁₆
christian	lol, ..., god ₁₂ , pray ₁₃ , ..., bless ₁₇ , ..., jesus ₂₀
dancer	dance, since, hey, never, been
directioner	harry, d, follow, direction, never, liam, niall
doctor	sweet, oh, or, life, nothing
engineer	(, then, since, may,), test ₉ , - ₁₇ , = ₁₈
freshman	summer, homework, na, ..., party ₁₉ , school ₂₀
geek	trying, oh, different, dead, been
grandma	morning, baby, around, night, excited
hipster	fucking, actually, thing, fuck, song
lawyer	did, never, his, may, pretty, law, even, office
man	man, away, ai, young, since
mother	morning, take, fuck, fucking, trying
nurse	lol, been, morning, ..., night ₁₀ , nursing ₁₁ , shift ₁₃
optimist	morning, enough, those, everything, never
poet	feel, song, even, say, yo
rapper	fuck, morning, lol, ..., mixtape ₈ , songs ₁₅
singer	sing, song, music, lol, never
smoker	fuck, shit, fucking, since, ass, smoke, weed ₂₀
solider	ai, beautiful, lol, wan, trying
sophomore	summer, >, ..., school ₁₁ , homework ₁₂
student	anything, summer, morning, since, actually
teacher	teacher, morning, teach, ..., students ₇ , ..., school ₂₀
vegetarian	actually, dead, summer, oh, morning
waitress	man, try, goes, hate, fat
woman	lol, into, woman, morning, never
writer	write, story, sweet, very, working

Table 3: Most-positively weighted features per role, along with select features within the top 20. Surprising **mother** features come from ambiguous self-identification, as seen in tweets such as: *I'm a mother f!cking starrrrr.*

ment. These results qualitatively suggest many roles under consideration may be teased out from a background population by focussing on language that follows expected use patterns. For example the use of the term *game* by athletes, *studio* by artists, *mixtape* by rappers, or *jesus* by Christians.

3 Characteristic Attributes

Bergsma and Van Durme (2013) showed that the

task of mining attributes for conceptual classes can relate straightforwardly to author attribute prediction. If one views a role, in their case gender, as two conceptual classes, **male** and **female**, then existing attribute extraction methods for third-person content (e.g., news articles) can be cheaply used to create a set of bootstrapping features for building classifiers over first-person content (e.g., tweets). For example, if we learn from news corpora that: *a man may have a wife*, then a tweet saying: *...my wife...* can be taken as potential evidence of membership in the **male** conceptual class.

In our second study, we test whether this idea extends to our wider set of fine-grained roles. For example, we aimed to discover that a **doctor** may *have a patient*, while a **hairdresser** may *have a salon*; these properties can be expressed in first-person content as possessives like *my patient* or *my salon*. We approached this task by selecting target roles from the first experiment and ranking characteristic attributes for each using pointwise mutual information (PMI) (Church and Hanks, 1990).

First, we counted all terms matching a target social role’s possessive pattern (e.g., *doctor’s*) in the web-scale n-gram corpus Google V2 (Lin et al., 2010)⁵. We ranked the collected terms by computing PMI between classes and attribute terms. Probabilities were estimated from counts of the class-attribute pairs along with counts matching the generic possessive patterns *his* and *her* which serve as general background categories. Following suggestions by Bergsma and Van Durme, we manually filtered the ranked list.⁶ We removed attributes that were either (a) not nominal, or (b) not indicative of the social role. This left fewer than 30 attribute terms per role, with many roles having fewer than 10.

We next performed a precision test to identify potentially useful attributes in these lists. We examined tweets with a first person possessive pattern for each attribute term from a small corpus of tweets collected over a single month in 2013, discarding those attribute terms with no positive matches. This precision test is useful regardless of how attribute lists are generated. The attribute

⁵In this corpus, follower-type roles like **belieber** and **directioner** are not at all prevalent. We therefore focused on occupational and habitual roles (e.g., **doctor**, **smoker**).

⁶Evidence from cognitive work on memory-dependent tasks suggests that such relevance based filtering (recognition) involves less cognitive effort than generating relevant attributes (recall) see (Jacoby et al., 1979). Indeed, this filtering step generally took less than a minute per class.

term *chart*, for example, had high PMI with **doctor**; but a precision test on the phrase *my chart* yielded a single tweet which referred not to a medical chart but to a top ten list (prompting removal of this attribute). Using this smaller high-precision set of attribute terms, we collected tweets from the Twitter Firehose over the period 2011-2013.

4 Attribute-based Classification

Attribute terms are less indicative overall than self-ID, e.g., the phrase *I’m a barber* is a clearer signal than *my scissors*. We therefore include a role verification step in curating a collection of positively identified users. We use the crowdsourcing platform Mechanical Turk⁷ to judge whether the person tweeting held a given role. Tweets were judged 5-way redundantly. Mechanical Turk judges (“Turkers”) were presented with a tweet and the prompt: *Based on this tweet, would you think this person is a **BARBER/HAIRDRESSER**?* along with four response options: *Yes, Maybe, Hard to tell, and No*.

We piloted this labeling task on 10 tweets per attribute term over a variety of classes. Each answer was associated with a score (Yes = 1, Maybe = .5, Hard to tell = No = 0) and aggregated across the five judges. We found in development that an aggregate score of 4.0 (out of 5.0) led to an acceptable agreement rate between the Turkers and the experimenters, when the tweets were randomly sampled and judged internally. We found that making conceptual class assignments based on a single tweet was often a subtle task. The results of this labeling study are shown in Figure 4, which gives the percent of tweets per attribute that were 4.0 or above. Attribute terms shown in red were manually discarded as being inaccurate (low on the y-axis) or non-prevalent (small shape).

From the remaining attribute terms, we identified users with tweets scoring 4.0 or better as positive examples of the associated roles. Tweets from those users were scraped via the Twitter API to construct corpora for each role. These were split into train and test, balanced with data from the same background set used in the self-ID study.

Test sets were usually of size 40 (20 positive, 20 background), with a few classes being sparse (the smallest had only 16 instances). Results are shown in Figure 5. Several classes in this balanced setup can be predicted with accuracies in the 70-90%

⁷<https://www.mturk.com/mturk/>

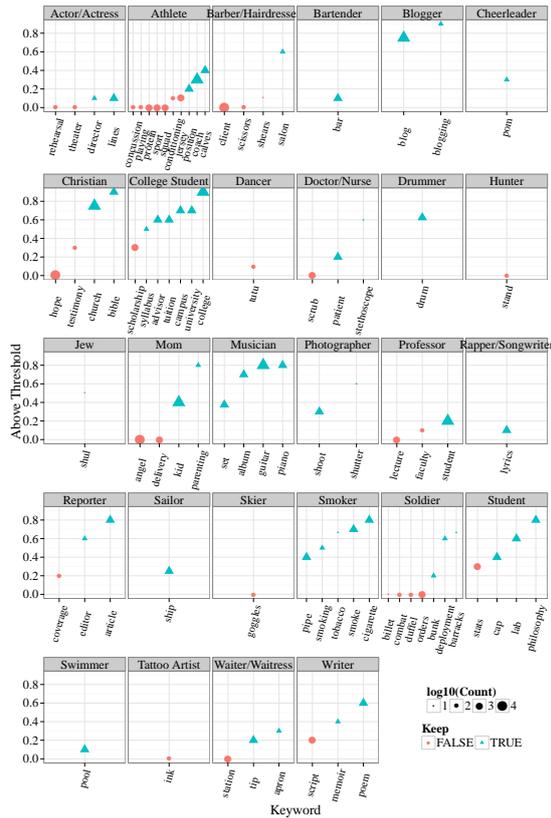


Figure 4: Turker judged quality of attributes selected as candidate features for bootstrapping positive instances of the given social role.

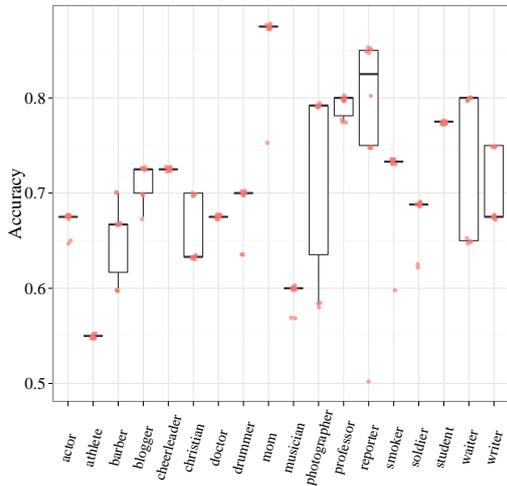


Figure 5: Classifier accuracy on balanced set contrasting agreed upon Twitter users of a given role against users pulled at random from the 1% stream.

range, supporting our claim that there is discriminating content for a variety of these social roles.

Conditional Classification How accurately we can predict membership in a given class when a Twitter user sends a tweet matching one of the targeted attributes? For example, if one sends a tweet saying *my coach*, then how likely is it that author

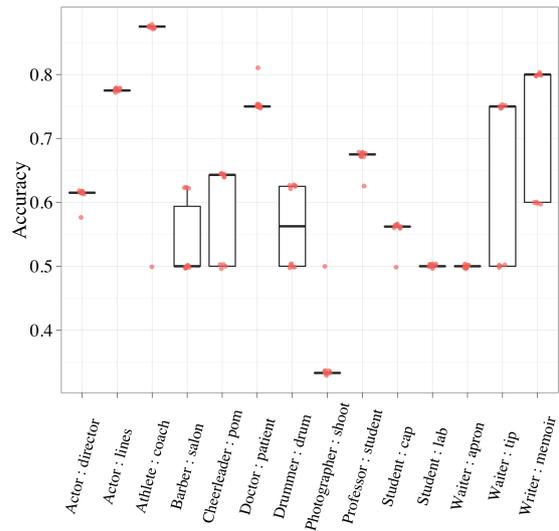


Figure 6: Results of positive vs negative by attribute term. Given that a user tweets ... *my lines* ... we are nearly 80% accurate in identifying whether or not the user is an actor.

is an **athlete**?

Using the same collection as the previous experiment, we trained classifiers conditioned on a given attribute term. Positive instances were taken to be those with a score of 4.0 or higher, with negative instances taken to be those with scores of 1.0 or lower (strong agreement by judges that the original tweet did not provide evidence of the given role). Classification results are shown in Figure 6.

5 Conclusion

We have shown that Twitter contains sufficiently robust signal to support more fine-grained author attribute prediction tasks than have previously been attempted. Our results are based on simple, intuitive search patterns with minimal additional filtering: this establishes the feasibility of the task, but leaves wide room for future work, both in the sophistication in methodology as well as the diversity of roles to be targeted. We exploited two complementary types of indicators: *self-identification* and *self-possession* of conceptual class (role) attributes. Those interested in identifying latent demographics can extend and improve these indicators in developing ways to identify groups of interest within the general population of Twitter users.

Acknowledgements This material is partially based on research sponsored by the NSF under grants DGE-123285 and IIS-1249516 and by DARPA under agreement number FA8750-13-2-0017 (the DEFT program).

References

- Abdulrahman Almuhareb and Massimo Poesio. 2004. Attribute-based and value-based clustering: an evaluation. In *Proceedings of EMNLP*.
- Shane Bergsma and Benjamin Van Durme. 2013. Using Conceptual Class Attributes to Characterize Social Media Users. In *Proceedings of ACL*.
- Shane Bergsma, Paul McNamee, Mossaab Bagdouri, Clay Fink, and Theresa Wilson. 2012. Language identification for creating language-specific twitter collections. In *Proceedings of the NAACL Workshop on Language and Social Media*.
- John D. Burger, John Henderson, George Kim, and Guido Zarrella. 2011. Discriminating gender on twitter. In *Proceedings of EMNLP*.
- Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.
- Michael Conover, Jacob Ratkiewicz, Matthew Francisco, Bruno Gonçalves, Filippo Menczer, and Alessandro Flammini. 2011. Political polarization on twitter. In *ICWSM*.
- Jacob Eisenstein, Brendan O’Connor, Noah Smith, and Eric P. Xing. 2010. A latent variable model of geographical lexical variation. In *Proceedings of EMNLP*.
- Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsief, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, (9).
- Nikesh Garera and David Yarowsky. 2009. Modeling latent biographic attributes in conversational genres. In *Proceedings of ACL*.
- Larry L Jacoby, Fergus IM Craik, and Ian Begg. 1979. Effects of decision difficulty on recognition and recall. *Journal of Verbal Learning and Verbal Behavior*, 18(5):585–600.
- Michal Kosinski, David Stillwell, and Thore Graepel. 2013. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*.
- Alex Lamb, Michael J. Paul, and Mark Dredze. 2013. Separating fact from fear: Tracking flu infections on twitter. In *Proceedings of NAACL*.
- Dekang Lin, Kenneth Church, Heng Ji, Satoshi Sekine, David Yarowsky, Shane Bergsma, Kailash Patil, Emily Pitler, Rachel Lathbury, Vikram Rao, Kapil Dalwani, and Sushant Narsale. 2010. New tools for web-scale n-grams. In *Proc. LREC*, pages 2221–2227.
- Saif M. Mohammad, Svetlana Kiritchenko, and Joel Martin. 2013. Identifying purpose behind electoral tweets. In *Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining*, WISDOM ’13, pages 1–9.
- Dong Nguyen, Noah A Smith, and Carolyn P Rosé. 2011. Author age prediction from text using linear regression. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 115–123. Association for Computational Linguistics.
- Marius Paşca and Benjamin Van Durme. 2008. Weakly-Supervised Acquisition of Open-Domain Classes and Class Attributes from Web Documents and Query Logs. In *Proceedings of ACL*.
- Michael J Paul and Mark Dredze. 2011. You are what you tweet: Analyzing twitter for public health. In *ICWSM*.
- Marco Pennacchiotti and Ana-Maria Popescu. 2011. Democrats, Republicans and Starbucks aficionados: User classification in Twitter. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, pages 430–438. ACM.
- Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. 2010. Classifying latent user attributes in twitter. In *Proceedings of the Workshop on Search and Mining User-generated Contents (SMUC)*.
- Lenhart K. Schubert. 2002. Can we derive general world knowledge from texts? In *Proceedings of HLT*.
- Benjamin Van Durme. 2012a. Jerboa: A toolkit for randomized and streaming algorithms. Technical Report 7, Human Language Technology Center of Excellence, Johns Hopkins University.
- Benjamin Van Durme. 2012b. Streaming analysis of discourse participants. In *Proceedings of EMNLP*.
- Jennifer Wortman. 2008. Viral marketing and the diffusion of trends on social networks. Technical Report MS-CIS-08-19, University of Pennsylvania, May.
- Faiyaz Al Zamal, Wendy Liu, and Derek Ruths. 2012. Homophily and latent attribute inference: Inferring latent attributes of Twitter users from neighbors. In *Proceedings of ICWSM*.

Automatically Detecting Corresponding Edit-Turn-Pairs in Wikipedia

Johannes Daxenberger[†] and Iryna Gurevych^{†‡}

[†] Ubiquitous Knowledge Processing Lab
Department of Computer Science, Technische Universität Darmstadt

[‡] Information Center for Education
German Institute for Educational Research and Educational Information

<http://www.ukp.tu-darmstadt.de>

Abstract

In this study, we analyze links between edits in Wikipedia articles and turns from their discussion page. Our motivation is to better understand implicit details about the writing process and knowledge flow in collaboratively created resources. Based on properties of the involved edit and turn, we have defined constraints for corresponding edit-turn-pairs. We manually annotated a corpus of 636 corresponding and non-corresponding edit-turn-pairs. Furthermore, we show how our data can be used to automatically identify corresponding edit-turn-pairs. With the help of supervised machine learning, we achieve an accuracy of .87 for this task.

1 Introduction

The process of user interaction in collaborative writing has been the topic of many studies in recent years (Erkens et al., 2005). Most of the resources used for collaborative writing do not explicitly allow their users to interact directly, so that the implicit effort of coordination behind the actual writing is not documented. Wikipedia, as one of the most prominent collaboratively created resources, offers its users a platform to coordinate their writing, the so called talk or discussion pages (Viégas et al., 2007). In addition to that, Wikipedia stores all edits made to any of its pages in a revision history, which makes the actual writing process explicit. We argue that linking these two resources helps to get a better picture of the collaborative writing process. To enable such interaction, we extract segments from discussion pages, called turns, and connect them to corresponding edits in the respective article. Consider the following snippet from the discussion page of the article “Boron”

in the English Wikipedia. On February 16th of 2011, user JCM83 added the **turn**:

Shouldn't borax be wikilinked in the
“etymology” paragraph?

Roughly five hours after that turn was issued on the discussion page, user Sbharris added a wikilink to the “History and etymology” section of the article by performing the following **edit**:

' ' borax ' ' → [[borax]]

This is what we define as a corresponding *edit-turn-pair*. More details follow in Section 2. To the best of our knowledge, this study is the first attempt to detect corresponding edit-turn-pairs in the English Wikipedia fully automatically.

Our motivation for this task is two-fold. First, an automatic detection of corresponding edit-turn-pairs in Wikipedia pages might help users of the encyclopedia to better understand the development of the article they are reading. Instead of having to read through all of the discussion page which can be an exhausting task for many of the larger articles in the English Wikipedia, users could focus on those discussions that actually had an impact on the article they are reading. Second, assuming that edits often introduce new knowledge to an article, it might be interesting to analyze how much of this knowledge was actually generated within the discourse on the discussion page.

The detection of correspondence between edits and turns is also relevant beyond Wikipedia. Many companies use Wikis to store internal information and documentation (Arazy et al., 2009). An alignment between edits in the company Wiki and issues discussed in email conversations, on mailing lists, or other forums, can be helpful to track the flow or generation of knowledge within the company. This information can be useful to improve communication and knowledge sharing.

In the limited scope of this paper, we will focus on two research questions. First, we want to understand the nature of correspondence between Wikipedia article edits and discussion page turns. Second, we want to know the distinctive properties of corresponding edit-turn-pairs and how to use these to automatically detect corresponding pairs.

2 Edit-Turn-Pairs

In this section, we will define the basic units of our task, namely edits and turns. Furthermore, we will explain the kind of correspondence between edits and turns we are interested in.

Edits To capture a fine-grained picture of changes to Wikipedia article pages, we rely on the notion of edits defined in our previous work (Daxenberger and Gurevych, 2012). Edits are coherent modifications based on a pair of adjacent revisions from Wikipedia article pages. To calculate edits, a line-based diff comparison between the old revision and the new revision is made, followed by several post-processing steps. Each pair of adjacent revisions found in the edit history of an article consists of one or more edits, which describe either inserted, deleted, changed or relocated text. Edits are associated with metadata from the revision they belong to, this includes the comment (if present), the user name and the time stamp.

Turns Turns are segments from Wikipedia discussion pages. To segment discussion pages into turns, we follow a procedure proposed by Ferschke et al. (2012). With the help of the Java Wikipedia Library (Zesch et al., 2008), we access discussion pages from a database. Discussion pages are then segmented into *topics* based upon the structure of the page. Individual turns are retrieved from topics by considering the revision history of the discussion page. This procedure successfully segmented 94 % of all turns in a corpus from the Simple English Wikipedia (Ferschke et al., 2012). Along with each turn, we store the name of its user, the time stamp, and the name of the topic to which the turn belongs.

Corresponding Edit-Turn-Pairs An edit-turn-pair is defined as a pair of an edit from a Wikipedia article’s revision history and a turn from the discussion page bound to the same article. If an article has no discussion page, there are no edit-turn-pairs for this article.

A definition of correspondence is not straightforward in the context of edit-turn-pairs. Ferschke et al. (2012) suggest four types of explicit performatives in their annotation scheme for dialog acts of Wikipedia turns. Due to their performative nature, we assume that these dialog acts make the turn they belong to a good candidate for a corresponding edit-turn-pair. We therefore define an edit-turn-pair as corresponding, if: i) The turn is an *explicit suggestion, recommendation or request* and the edit performs this suggestion, recommendation or request, ii) the turn is an *explicit reference or pointer* and the edit adds or modifies this reference or pointer, iii) the turn is a *commitment to an action in the future* and the edit performs this action, and iv) the turn is a *report of a performed action* and the edit performs this action. We define all edit-turn-pairs which do not conform to the upper classification as non-corresponding.

3 Corpus

With the help of Amazon Mechanical Turk¹, we crowdsourced annotations on a corpus of edit-turn-pairs from 26 random English Wikipedia articles in various thematic categories. The search space for corresponding edit-turn-pairs is quite big, as any edit to an article may correspond to any turn from the article’s discussion page. Assuming that most edit-turn-pairs are non-corresponding, we expect a heavy imbalance in the class distribution. It was important to find a reasonable amount of corresponding edit-turn-pairs before the actual annotation could take place, as we needed a certain amount of positive seeds to keep turkers from simply labeling pairs as non-corresponding all the time. In the following, we explain the step-by-step approach we chose to create a suitable corpus for the annotation study.

Filtering We applied various filters to avoid annotating trivial content. Based on an automatic classification using the model presented in our previous work (Daxenberger and Gurevych, 2013), we excluded edits classified as Vandalism, Revert or Other. Furthermore, we removed all edits which are part of a revision created by bots, based on the Wikimedia user group² scheme. To keep the class imbalance within reasonable margins, we limited the time span between edits and turns to 86,000

¹www.mturk.com

²http://meta.wikimedia.org/wiki/User_classes

seconds (about 24 hours). The result is a set of 13,331 edit-turn-pairs, referred to as *ETP-all*.

Preliminary Annotation Study From *ETP-all*, a set of 262 edit-turn-pairs have been annotated as corresponding as part of a preliminary annotation study with one human annotator. This step is intended to make sure that we have a substantial number of corresponding pairs in the data for the final annotation study. However, we still expect a certain amount of non-corresponding edit-turn-pairs in this data, as the annotator judged the correspondence based on the entire revision and not the individual edit. We refer to this 262 edit-turn-pairs as *ETP-unconfirmed*.

Mechanical Turk Annotation Study Finally, for the Mechanical Turk annotation study, we selected 500 random edit-turn-pairs from *ETP-all* excluding *ETP-unconfirmed*. Among these, we expect to find mostly non-corresponding pairs. From *ETP-unconfirmed*, we selected 250 random edit-turn-pairs. The resulting 750 pairs have each been annotated by five turkers. The turkers were presented the turn text, the turn topic name, the edit in its context, and the edit comment (if present). The context of an edit is defined as one preceding and one following paragraph of the edited paragraph. Each edit-turn-pair could be labeled as “corresponding”, “non-corresponding” or “can’t tell”. To select good turkers and to block spammers, we carried out a pilot study on a small portion of manually confirmed corresponding and non-corresponding pairs, and required turkers to pass a qualification test.

The average pairwise percentage agreement over all pairs is 0.66. This was calculated as $\frac{1}{N} \sum_{i=1}^N \frac{\sum_{c=1}^C v_i^c}{C}$, where $N = 750$ is the overall number of annotated edit-turn-pairs, $C = \frac{R^2 - R}{2}$ is the number of pairwise comparisons, $R = 5$ is the number of raters per edit-turn-pair, and $v_i^c = 1$ if a pair of raters c labeled edit-turn-pair i equally, and 0 otherwise. The moderate pairwise agreement reflects the complexity of this task for non-experts.

Gold Standard To rule out ambiguous cases, we created the Gold Standard corpus with the help of majority voting. We counted an edit-turn-pair as corresponding, if it was annotated as “corresponding” by least three out of five annotators, and likewise for non-corresponding pairs. Furthermore, we deleted 21 pairs for which the turn seg-

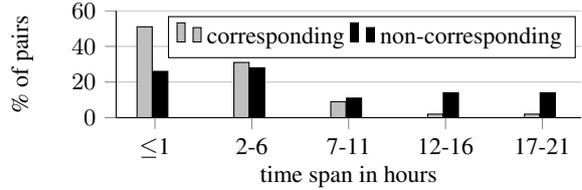


Figure 1: Percentage of (non-)corresponding edit-turn-pairs for various time intervals in *ETP-gold*.

mentation algorithm clearly failed (e.g. when the turn text was empty). This resulted in 128 corresponding and 508 non-corresponding pairs, or 636 pairs in total. We refer to this dataset as *ETP-gold*. To assess the reliability of these annotations, one of the co-authors manually annotated a random subset of 100 edit-turn-pairs contained in *ETP-gold* as corresponding or non-corresponding. The inter-rater agreement between *ETP-gold* (majority votes over Mechanical Turk annotations) and our expert annotations on this subset is Cohen’s $\kappa = .72$. We consider this agreement high enough to draw conclusions from the annotations (Artstein and Poesio, 2008).

Obviously, this is a fairly small dataset which does not cover a representative sample of articles from the English Wikipedia. However, given the high price for a new corresponding edit-turn-pair (due to the high class imbalance in random data), we consider it as a useful starting point for research on edit-turn-pairs in Wikipedia. We make *ETP-gold* freely available.³

As shown in Figure 1, more than 50% of all corresponding edit-turn-pairs in *ETP-gold* occur within a time span of less than one hour. In our 24 hours search space, the probability to find a corresponding edit-turn-pair drops steeply for time spans of more than 6 hours. We therefore expect to cover the vast majority of corresponding edit-turn-pairs within a search space of 24 hours.

4 Machine Learning with Edit-Turn-Pairs

We used DKPro TC (Daxenberger et al., 2014) to carry out the machine learning experiments on edit-turn-pairs. For each edit, we stored both the edited paragraph and its context from the old revision as well as the edited paragraph and context from the new revision. We used Apache

³<http://www.ukp.tu-darmstadt.de/data/edit-turn-pairs>

OpenNLP⁴ for the segmentation of edit and turn text. Training and testing the classifier has been carried out with the help of the Weka Data Mining Software (Hall et al., 2009). We used the Sweble parser (Dohrn and Riehle, 2011) to remove Wiki markup.

4.1 Features

In the following, we list the features extracted from preprocessed edits and turns. The *edit text* is composed of any inserted, deleted or relocated text from both the old and the new revision. The *edit context* includes the edited paragraph and one preceding and one following paragraph. The *turn text* includes the entire text from the turn.

Similarity between turn and edit text We propose a number of features which are purely based on the textual similarity between the text of the turn, and the edited text and context. We used the cosine similarity, longest common subsequence, and word n-gram similarity measures. Cosine similarity was applied on binary weighted term vectors (L^2 norm). The word n-gram measure (Lyon et al., 2004) calculates a Jaccard similarity coefficient on trigrams. Similarity has been calculated between i) the plain edit text and the turn text, ii) the edit and turn text after any wiki markup has been removed, iii) the plain edit context and turn text, and iv) the edit context and turn text after any wiki markup has been removed.

Based on metadata of edit and turn Several of our features are based on metadata from both the edit and the turn. We recorded whether the name of the edit user and the turn user are equal, the absolute time difference between the turn and the edit, and whether the edit occurred before the turn. Cosine similarity, longest common subsequence, and word n-gram similarity were also applied to measure the similarity between the edit comment and the turn text as well as the similarity between the edit comment and the turn topic name.

Based on either edit or turn Some features are based on the edit or the turn alone and do not take into account the pair itself. We recorded whether the edit is an insertion, deletion, modification or relocation. Furthermore, we measured the length of the edit text and the length of the turn text. The 1,000 most frequent uni-, bi- and trigrams from the turn text are represented as binary features.

⁴<http://opennlp.apache.org>

	Baseline	R. Forest	SVM
Accuracy	.799 \pm .031	.866 \pm .026 [†]	.858 \pm .027 [†]
F1 _{mac.}	NaN	.789 \pm .032	.763 \pm .033
Precision _{mac.}	NaN	.794 \pm .031	.791 \pm .032
Recall _{mac.}	.500 \pm .039	.785 \pm .032 [†]	.736 \pm .034 [†]
F1 _{non-corr.}	.888 \pm .025	.917 \pm .021	.914 \pm .022
F1 _{corr.}	NaN	.661 \pm .037	.602 \pm .038

Table 1: Classification results from a 10-fold cross-validation experiment on ETP-gold with 95% confidence intervals. Non-overlapping intervals w.r.t. the majority baseline are marked by [†].

4.2 Classification Experiments

We treat the automatic classification of edit-turn-pairs as a binary classification problem. Given the small size of ETP-gold, we did not assign a fixed train/test split to the data. For the same reason, we did not further divide the data into train/test and development data. Rather, hyperparameters were optimized using grid-search over multiple cross-validation experiments, aiming to maximize accuracy. To deal with the class imbalance problem, we applied cost-sensitive classification. In correspondence with the distribution of class sizes in the training data, the cost for false negatives was set to 4, and for false positives to 1. A reduction of the feature set as judged by a χ^2 ranker improved the results for both Random Forest as well as the SVM, so we limited our feature set to the 100 best features.

In a 10-fold cross-validation experiment, we tested a Random Forest classifier (Breiman, 2001) and an SVM (Platt, 1998) with polynomial kernel. Previous work (Ferschke et al., 2012; Bronner and Monz, 2012) has shown that these algorithms work well for edit and turn classification. As baseline, we defined a majority class classifier, which labels all edit-turn-pairs as non-corresponding.

4.3 Discussion and Error Analysis

The classification results for the above configuration are displayed in Table 1. Due to the high class imbalance in the data, the majority class baseline sets a challenging accuracy score of .80. Both classifiers performed significantly better than the baseline (non-overlapping confidence intervals, see Table 1). With an overall macro-averaged F1 of .79, Random Forest yielded the best results, both with respect to precision as well as recall. The low F1 on corresponding pairs is likely due to the small number of training examples.

To understand the mistakes of the classifier, we manually assessed error patterns within the model of the Random Forest classifier. Some of the false positives (i.e. non-corresponding pairs classified as corresponding) were caused by pairs where the revision (as judged by its comment or the edit context) is related to the turn text, however the specific edit in this pair is not. This might happen, when somebody corrects a spelling error in a paragraph that is heavily disputed on the discussion page. Among the false negatives, we found errors caused by a missing direct textual overlap between edit and turn text. In these cases, the correspondence was indicated only (if at all) by some relationship between turn text and edit comment.

5 Related Work

Besides the work by Ferschke et al. (2012) which is the basis for our turn segmentation, there are several studies dedicated to discourse structure in Wikipedia. Viégas et al. (2007) propose 11 dimensions to classify discussion page turns. The most frequent dimensions in their sample are requests for coordination and requests for information. Both of these may be part of a corresponding edit-turn-pair, according to our definition in Section 2. A subsequent study (Schneider et al., 2010) adds more dimensions, among these an explicit category for references to article edits. This dimension accounts for roughly 5 to 10% of all turns. Kittur and Kraut (2008) analyze correspondence between article quality and activity on the discussion page. Their study shows that both implicit coordination (on the article itself) and explicit coordination (on the discussion page of the article) play important roles for the improvement of article quality. In the present study, we have analyzed cases where explicit coordination lead to implicit coordination and vice versa.

Kaltenbrunner and Laniado (2012) analyze the development of discussion pages in Wikipedia with respect to time and compare dependences between edit peaks in the revision history of the article itself and the respective discussion page. They find that the development of a discussion page is often bound to the topic of the article, i.e. articles on time-specific topics such as events grow much faster than discussions about timeless, encyclopedic content. Furthermore, they observed that the edit peaks in articles and their discussion pages are mostly independent. This partially explains the

high number of non-corresponding edit-turn-pairs and the consequent class imbalance.

While there are several studies which analyze the high-level relationship between discussion and edit activity in Wikipedia articles, very few have investigated the correspondence between edits and turns on the textual level. Among the latter, Ferron and Massa (2014) analyze 88 articles and their discussion pages related to traumatic events. In particular, they find a correlation between the article edits and their discussions around the anniversaries of the events.

6 Conclusion

The novelty of this paper is a computational analysis of the relationship between the edit history and the discussion of a Wikipedia article. As far as we are aware, this is the first study to automatically analyze this relationship involving the textual content of edits and turns. Based on the types of turn and edit in an edit-turn-pair, we have operationalized the notion of corresponding and non-corresponding edit-turn-pairs. The basic assumption is that in a corresponding pair, the turn contains an explicit performative and the edit corresponds to this performative. We have presented a machine learning system to automatically detect corresponding edit-turn-pairs. To test this system, we manually annotated a corpus of corresponding and non-corresponding edit-turn-pairs. Trained and tested on this data, our system shows a significant improvement over the baseline.

With regard to future work, an extension of the manually annotated corpus is the most important issue. Our classifier can be used to bootstrap the annotation of additional edit-turn-pairs.

Acknowledgments

The authors would like to give special thanks to Viswanathan Arunachalam and Dat Quoc Nguyen, who carried out initial experiments and the preliminary annotation study, and to Emily Jamison, who set up the Mechanical Turk task. This work has been supported by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806, and by the Hessian research excellence program “Landes-Offensive zur Entwicklung Wissenschaftlich-ökonomischer Exzellenz” (LOEWE) as part of the research center “Digital Humanities”. We thank the anonymous reviewers for their helpful suggestions.

References

- Ofner Arazy, Ian Gellatly, Soobaeck Jang, and Raymond Patterson. 2009. Wiki deployment in corporate settings. *IEEE Technology and Society Magazine*, 28(2):57–64.
- Ron Artstein and Massimo Poesio. 2008. Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, 34(4):555–596.
- Leo Breiman. 2001. Random Forests. *Machine Learning*, 45(1):5–32.
- Amit Bronner and Christof Monz. 2012. User Edits Classification Using Document Revision Histories. In *European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 356–366, Avignon, France.
- Johannes Daxenberger and Iryna Gurevych. 2012. A Corpus-Based Study of Edit Categories in Featured and Non-Featured Wikipedia Articles. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 711–726, Mumbai, India.
- Johannes Daxenberger and Iryna Gurevych. 2013. Automatically Classifying Edit Categories in Wikipedia Revisions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 578–589, Seattle, WA, USA.
- 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 the 52nd Annual Meeting of the Association for Computational Linguistics. System Demonstrations*, page (to appear), Baltimore, MD, USA.
- Hannes Dohrn and Dirk Riehle. 2011. Design and implementation of the Sweble Wikitext parser. In *Proceedings of the International Symposium on Wikis and Open Collaboration (WikiSym '11)*, pages 72–81, Mountain View, CA, USA.
- Gijsbert Erkens, Jos Jaspers, Maaike Prangma, and Gellof Kanselaar. 2005. Coordination processes in computer supported collaborative writing. *Computers in Human Behavior*, 21(3):463–486.
- Michela Ferron and Paolo Massa. 2014. Beyond the encyclopedia: Collective memories in Wikipedia. *Memory Studies*, 7(1):22–45.
- Oliver Ferschke, Iryna Gurevych, and Yevgen Chebotar. 2012. Behind the Article: Recognizing Dialog Acts in Wikipedia Talk Pages. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 777–786, Avignon, France.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18.
- Andreas Kaltenbrunner and David Laniado. 2012. There is No Deadline - Time Evolution of Wikipedia Discussions. In *Proceedings of the Annual International Symposium on Wikis and Open Collaboration*, Linz, Austria.
- Aniket Kittur and Robert E. Kraut. 2008. Harnessing the wisdom of crowds in wikipedia: quality through coordination. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, pages 37–46, San Diego, CA, USA.
- C. Lyon, R. Barrett, and J. Malcolm. 2004. A theoretical basis to the automated detection of copying between texts, and its practical implementation in the Ferret plagiarism and collusion detector. In *Plagiarism: Prevention, Practice and Policy Conference*, Newcastle, UK.
- John C. Platt. 1998. Fast training of support vector machines using sequential minimal optimization. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 185–208. MIT Press.
- Jodi Schneider, Alexandre Passant, and John G. Breslin. 2010. A Content Analysis: How Wikipedia Talk Pages Are Used. In *Proceedings of the 2nd International Conference of Web Science*, pages 1–7, Raleigh, NC, USA.
- Fernanda B. Viégas, Martin Wattenberg, Jesse Kriss, and Frank Ham. 2007. Talk Before You Type: Coordination in Wikipedia. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, pages 78–78, Big Island, HI, USA.
- Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco.

Two Knives Cut Better Than One: Chinese Word Segmentation with Dual Decomposition

Mengqiu Wang	Rob Voigt	Christopher D. Manning
Computer Science Department	Linguistics Department	Computer Science Department
Stanford University	Stanford University	Stanford University
Stanford, CA 94305	Stanford, CA 94305	Stanford, CA 94305
{mengqiu, manning}@cs.stanford.edu		robvoigt@stanford.edu

Abstract

There are two dominant approaches to Chinese word segmentation: word-based and character-based models, each with respective strengths. Prior work has shown that gains in segmentation performance can be achieved from combining these two types of models; however, past efforts have not provided a practical technique to allow mainstream adoption. We propose a method that effectively combines the strength of both segmentation schemes using an efficient dual-decomposition algorithm for joint inference. Our method is simple and easy to implement. Experiments on SIGHAN 2003 and 2005 evaluation datasets show that our method achieves the best reported results to date on 6 out of 7 datasets.

1 Introduction

Chinese text is written without delimiters between words; as a result, Chinese word segmentation (CWS) is an essential foundational step for many tasks in Chinese natural language processing. As demonstrated by (Shi and Wang, 2007; Bai et al., 2008; Chang et al., 2008; Kummerfeld et al., 2013), the quality and consistency of segmentation has important downstream impacts on system performance in machine translation, POS tagging and parsing.

State-of-the-art performance in CWS is high, with F-scores in the upper 90s. Still, challenges remain. Unknown words, also known as out-of-vocabulary (OOV) words, lead to difficulties for word- or dictionary-based approaches. Ambiguity can cause errors when the appropriate segmentation is determined contextually, such as 才能 (“talent”) and 才 / 能 (“just able”) (Gao et al., 2003).

There are two primary classes of models: character-based, where the foundational units for

processing are individual Chinese characters (Xue, 2003; Tseng et al., 2005; Zhang et al., 2006; Wang et al., 2010), and word-based, where the units are full words based on some dictionary or training lexicon (Andrew, 2006; Zhang and Clark, 2007). Sun (2010) details their respective theoretical strengths: character-based approaches better model the internal compositional structure of words and are therefore more effective at inducing new OOV words; word-based approaches are better at reproducing the words of the training lexicon and can capture information from significantly larger contextual spans. Prior work has shown performance gains from combining these two types of models to exploit their respective strengths, but such approaches are often complex to implement and computationally expensive.

In this work, we propose a simple and principled joint decoding method for combining character-based and word-based segmenters based on dual decomposition. This method has strong optimality guarantees and works very well empirically. It is easy to implement and does not require retraining of existing character- and word-based segmenters. Perhaps most importantly, this work presents a much more practical and usable form of classifier combination in the CWS context than existing methods offer.

Experimental results on standard SIGHAN 2003 and 2005 bake-off evaluations show that our model outperforms the character and word baselines by a significant margin. In particular, our approach improves OOV recall rates and segmentation consistency, and gives the best reported results to date on 6 out of 7 datasets.

2 Models for CWS

Here we describe the character-based and word-based models we use as baselines, review existing approaches to combination, and describe our algorithm for joint decoding with dual decomposition.

2.1 Character-based Models

In the most commonly used contemporary approach to character-based segmentation, first proposed by (Xue, 2003), CWS is seen as a character sequence tagging task, where each character is tagged on whether it is at the beginning, middle, or end of a word. Conditional random fields (CRF) (Lafferty et al., 2001) have been widely adopted for this task, and give state-of-the-art results (Tseng et al., 2005). In a first-order linear-chain CRF model, the conditional probability of a label sequence \mathbf{y} given a word sequence \mathbf{x} is defined as:

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z} \sum_{t=1}^{|\mathbf{y}|} \exp(\theta \cdot f(x, y_t, y_{t+1}))$$

$f(x, y_t, y_{t+1})$ are feature functions that typically include surrounding character n-gram and morphological suffix/prefix features. These types of features capture the compositional properties of characters and are likely to generalize well to unknown words. However, the Markov assumption in CRF limits the context of such features; it is difficult to capture long-range word features in this model.

2.2 Word-based Models

Word-based models search through lists of word candidates using scoring functions that directly assign scores to each. Early word-based segmentation work employed simple heuristics like dictionary-lookup maximum matching (Chen and Liu, 1992). More recently, Zhang and Clark (2007) reported success using a linear model trained with the average perceptron algorithm (Collins, 2002). Formally, given input \mathbf{x} , their model seeks a segmentation y such that:

$$F(\mathbf{y}|\mathbf{x}) = \max_{\mathbf{y} \in \text{GEN}(\mathbf{x})} (\alpha \cdot \phi(\mathbf{y}))$$

$F(\mathbf{y}|\mathbf{x})$ is the score of segmentation result y . Searching through the entire $\text{GEN}(\mathbf{x})$ space is intractable even with a local model, so a beam-search algorithm is used. The search algorithm consumes one character input token at a time, and iterates through the existing beams to score two new alternative hypotheses by either appending the new character to the last word in the beam, or starting a new word at the current position.

Algorithm 1 Dual decomposition inference algorithm, and modified Viterbi and beam-search algorithms.

```

 $\forall i \in \{1 \text{ to } |\mathbf{x}|\}: \forall k \in \{0, 1\}: u_i(k) = 0$ 
for  $t \leftarrow 1$  to  $T$  do
   $\mathbf{y}^{c*} = \operatorname{argmax}_{\mathbf{y}} P(\mathbf{y}^c|\mathbf{x}) + \sum_{i \in |\mathbf{x}|} u_i(y_i^c)$ 
   $\mathbf{y}^{w*} = \operatorname{argmax}_{\mathbf{y} \in \text{GEN}(\mathbf{x})} F(\mathbf{y}^w|\mathbf{x}) - \sum_{j \in |\mathbf{x}|} u_j(y_j^w)$ 
  if  $\mathbf{y}^{c*} = \mathbf{y}^{w*}$  then
    return  $(\mathbf{y}^{c*}, \mathbf{y}^{w*})$ 
  end if
  for all  $i \in \{1 \text{ to } |\mathbf{x}|\}$  do
     $\forall k \in \{0, 1\}: u_i(k) = u_i(k) + \alpha_t(2k - 1)(y_i^{w*} - y_i^{c*})$ 
  end for
end for
return  $(\mathbf{y}^{c*}, \mathbf{y}^{w*})$ 

```

```

Viterbi:
 $V_1(1) = 1, V_1(0) = 0$ 
for  $i = 2$  to  $|\mathbf{x}|$  do
   $\forall k \in \{0, 1\}: V_i(k) = \operatorname{argmax}_{k'} P_i(k|k') V_{i-1}k' + u_i(k)$ 
end for

```

```

Beam-Search:
for  $i = 1$  to  $|\mathbf{x}|$  do
  for item  $v = \{w_0, \dots, w_j\}$  in  $\text{beam}(i)$  do
    append  $x_i$  to  $w_j$ ,  $\text{score}(v) \stackrel{\pm}{=} u_i(0)$ 
     $v = \{w_0, \dots, w_j, x_i\}$ ,  $\text{score}(v) \stackrel{\pm}{=} u_i(1)$ 
  end for
end for

```

2.3 Combining Models with Dual Decomposition

Various mixing approaches have been proposed to combine the above two approaches (Wang et al., 2006; Lin, 2009; Sun et al., 2009; Sun, 2010; Wang et al., 2010). These mixing models perform well on standard datasets, but are not in wide use because of their high computational costs and difficulty of implementation.

Dual decomposition (DD) (Rush et al., 2010) offers an attractive framework for combining these two types of models without incurring high costs in model complexity (in contrast to (Sun et al., 2009)) or decoding efficiency (in contrast to bagging in (Wang et al., 2006; Sun, 2010)). DD has been successfully applied to similar situations for combining local with global models; for example, in dependency parsing (Koo et al., 2010), bilingual sequence tagging (Wang et al., 2013) and word alignment (DeNero and Macherey, 2011).

The idea is that jointly modelling both character-sequence and word information can be computationally challenging, so instead we can try to find outputs that the two models are most likely

	Academia Sinica					Peking Univ.				
	R	P	F ₁	R _{oov}	C	R	P	F ₁	R _{oov}	C
Char-based CRF	95.2	93.6	94.4	58.9	0.064	94.6	95.3	94.9	77.8	0.089
Word-based Perceptron	95.8	95.0	95.4	69.5	0.060	94.1	95.5	94.8	76.7	0.099
Dual-decomp	95.9	94.9	95.4	67.7	0.055	94.8	95.7	95.3	78.7	0.086
	City Univ. of Hong Kong					Microsoft Research				
	R	P	F ₁	R _{oov}	C	R	P	F ₁	R _{oov}	C
Char-based CRF	94.7	94.0	94.3	76.1	0.065	96.4	96.6	96.5	71.3	0.074
Word-based Perceptron	94.3	94.0	94.2	71.7	0.073	97.0	97.2	97.1	74.6	0.063
Dual-decomp	95.0	94.4	94.7	75.3	0.062	97.3	97.4	97.4	76.0	0.055

Table 1: Results on SIGHAN 2005 datasets. R_{oov} denotes OOV recall, and C denotes segmentation consistency. Best number in each column is highlighted in bold.

to agree on. Formally, the objective of DD is:

$$\max_{\mathbf{y}^c, \mathbf{y}^w} P(\mathbf{y}^c | \mathbf{x}) + F(\mathbf{y}^w | \mathbf{x}) \text{ s.t. } \mathbf{y}^c = \mathbf{y}^w \quad (1)$$

where \mathbf{y}^c is the output of character-based CRF, \mathbf{y}^w is the output of word-based perceptron, and the agreements are expressed as constraints. *s.t.* is a shorthand for “such that”.

Solving this constrained optimization problem directly is difficult. Instead, we take the Lagrangian relaxation of this term as:

$$L(\mathbf{y}^c, \mathbf{y}^w, \mathbf{U}) = P(\mathbf{y}^c | \mathbf{x}) + F(\mathbf{y}^w | \mathbf{x}) + \sum_{i \in |\mathbf{x}|} u_i (y_i^c - y_i^w) \quad (2)$$

where \mathbf{U} is the set of Lagrangian multipliers that consists of a multiplier u_i at each word position i .

We can rewrite the original objective with the Lagrangian relaxation as:

$$\max_{\mathbf{y}^c, \mathbf{y}^w} \min_{\mathbf{U}} L(\mathbf{y}^c, \mathbf{y}^w, \mathbf{U}) \quad (3)$$

We can then form the dual of this problem by taking the min outside of the max, which is an upper bound on the original problem. The dual form can then be decomposed into two sub-components (the two max problems in Eq. 4), each of which is local with respect to the set of Lagrangian multipliers:

$$\min_{\mathbf{U}} \left(\max_{\mathbf{y}^c} \left[P(\mathbf{y}^c | \mathbf{x}) + \sum_{i \in |\mathbf{x}|} u_i (y_i^c) \right] + \max_{\mathbf{y}^w} \left[F(\mathbf{y}^w | \mathbf{x}) - \sum_{j \in |\mathbf{x}|} u_j (y_j^w) \right] \right) \quad (4)$$

This method is called dual decomposition (DD) (Rush et al., 2010). Similar to previous work

(Rush and Collins, 2012), we solve this DD problem by iteratively updating the sub-gradient as depicted in Algorithm 1.¹ In each iteration, if the best segmentations provided by the two models do not agree, then the two models will receive penalties for the decisions they made that differ from the other. This penalty exchange is similar to message passing, and as the penalty accumulates over iterations, the two models are pushed towards agreeing with each other. We also give an updated Viterbi decoding algorithm for CRF and a modified beam-search algorithm for perceptron in Algorithm 1. T is the maximum number of iterations before early stopping, and α_t is the learning rate at time t . We adopt a learning rate update rule from Koo et al. (2010) where α_t is defined as $\frac{1}{N}$, where N is the number of times we observed a consecutive dual value increase from iteration 1 to t .

3 Experiments

We conduct experiments on the SIGHAN 2003 (Sproat and Emerson, 2003) and 2005 (Emerson, 2005) bake-off datasets to evaluate the effectiveness of the proposed dual decomposition algorithm. We use the publicly available Stanford CRF segmenter (Tseng et al., 2005)² as our character-based baseline model, and reproduce the perceptron-based segmenter from Zhang and Clark (2007) as our word-based baseline model.

We adopted the development setting from (Zhang and Clark, 2007), and used CTB sections 1-270 for training and sections 400-931 for development in hyper-parameter setting; for all results given in tables, the models are trained and evaluated on the standard train/test split for the given dataset. The optimized hyper-parameters used are:

¹See Rush and Collins (2012) for a full introduction to DD.

²<http://nlp.stanford.edu/software/segmenter.shtml>

ℓ_2 regularization parameter λ in CRF is set to 3; the perceptron is trained for 10 iterations with beam size 200; dual decomposition is run to max iteration of 100 (T in Algo. 1) with step size 0.1 (α_t in Algo. 1).

Beyond standard precision (P), recall (R) and F_1 scores, we also evaluate segmentation consistency as proposed by (Chang et al., 2008), who have shown that increased segmentation consistency is correlated with better machine translation performance. The consistency measure calculates the entropy of segmentation variations — the lower the score the better. We also report out-of-vocabulary recall (R_{OOV}) as an estimation of the model’s generalizability to previously unseen words.

4 Results

Table 1 shows our empirical results on SIGHAN 2005 dataset. Our dual decomposition method outperforms both the word-based and character-based baselines consistently across all four subsets in both F_1 and OOV recall (R_{OOV}). Our method demonstrates a robustness across domains and segmentation standards regardless of which baseline model was stronger. Of particular note is DD’s is much more robust in R_{OOV} , where the two baselines swing a lot. This is an important property for downstream applications such as entity recognition. The DD algorithm is also more consistent, which would likely lead to improvements in applications such as machine translation (Chang et al., 2008).

The improvement over our word- and character-based baselines is also seen in our results on the earlier SIGHAN 2003 dataset. Table 2 puts our method in the context of earlier systems for CWS. Our method achieves the best reported score on 6 out of 7 datasets.

5 Discussion and Error Analysis

On the whole, dual decomposition produces state-of-the-art segmentations that are more accurate, more consistent, and more successful at inducing OOV words than the baseline systems that it combines. On the SIGHAN 2005 test set, in over 99.1% of cases the DD algorithm converged within 100 iterations, which gives an optimality guarantee. In 77.4% of the cases, DD converged in the first iteration. The number of iterations to convergence histogram is plotted in Figure 1.

SIGHAN 2005				
	AS	PU	CU	MSR
<i>Best 05</i>	95.2	95.0	94.3	96.4
<i>Zhang et al. 06</i>	94.7	94.5	94.6	96.4
<i>Z&C 07</i>	94.6	94.5	95.1	97.2
<i>Sun et al. 09</i>	-	95.2	94.6	97.3
<i>Sun 10</i>	95.2	95.2	95.6	96.9
Dual-decomp	95.4	95.3	94.7	97.4
SIGHAN 2003				
<i>Best 03</i>	96.1	95.1	94.0	
<i>Peng et al. 04</i>	95.6	94.1	92.8	
<i>Z&C 07</i>	96.5	94.0	94.6	
Dual-decomp	97.1	95.4	94.9	

Table 2: Performance of dual decomposition in comparison to past published results on SIGHAN 2003 and 2005 datasets. Best reported F_1 score for each dataset is highlighted in bold. *Z&C 07* refers to Zhang and Clark (2007). *Best 03, 05* are results of the winning systems for each dataset in the respective shared tasks.

Error analysis In many cases the relative confidence of each model means that dual decomposition is capable of using information from both sources to generate a series of correct segmentations better than either baseline model alone. The example below shows a difficult-to-segment proper name comprised of common characters, which results in undersegmentation by the character-based CRF and oversegmentation by the word-based perceptron, but our method achieves the correct middle ground.

Gloss Tian Yage / ’s / creations
Gold 田雅各 / 的 / 创作
CRF 田雅各的 / 创作
PCPT 田雅 / 各 / 的 / 创作
DD 田雅各 / 的 / 创作

A powerful feature of the dual decomposition approach is that it can generate correct segmentation decisions in cases where a voting or product-of-experts model could not, since joint decoding allows the sharing of information at decoding time. In the following example, both baseline models miss the contextually clear use of the word 点心 (“sweets / snack food”) and instead attach 点 to the prior word to produce the otherwise common compound 一点点 (“a little bit”); dual decomposition allows the model to generate the correct segmentation.

Gloss Enjoy / a bit of / snack food / , ...
Gold 享受 / 一点 / 点心 / ,
CRF 享受 / 一点点 / 心 / ,
PCPT 享受 / 一点点 / 心 / ,
DD 享受 / 一点 / 点心 / ,

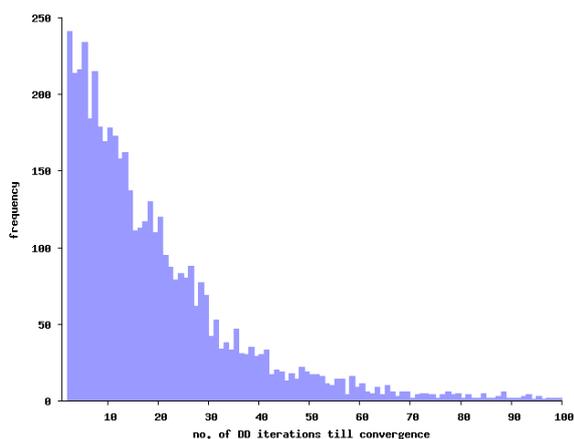


Figure 1: No. of iterations till DD convergence.

We found more than 400 such surprisingly accurate instances in our dual decomposition output.

Finally, since dual decomposition is a method of joint decoding, it is still liable to reproduce errors made by the constituent systems.

6 Conclusion

In this paper we presented an approach to Chinese word segmentation using dual decomposition for system combination. We demonstrated that this method allows for joint decoding of existing CWS systems that is more accurate and consistent than either system alone, and further achieves the best performance reported to date on standard datasets for the task. Perhaps most importantly, our approach is straightforward to implement and does not require retraining of the underlying segmentation models used. This suggests its potential for broader applicability in real-world settings than existing approaches to combining character-based and word-based models for Chinese word segmentation.

Acknowledgements

We gratefully acknowledge the support of the U.S. Defense Advanced Research Projects Agency (DARPA) Broad Operational Language Translation (BOLT) program through IBM. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, or the US government.

References

- Galen Andrew. 2006. A hybrid Markov/semi-Markov conditional random field for sequence segmentation. In *Proceedings of EMNLP*.
- Ming-Hong Bai, Keh-Jiann Chen, and Jason S. Chang. 2008. Improving word alignment by adjusting chinese word segmentation. In *Proceedings of the third International Joint Conference on Natural Language Processing (IJCNLP)*.
- Pichuan Chang, Michel Galley, and Chris Manning. 2008. Optimizing chinese word segmentation for machine translation performance. In *Proceedings of the ACL Workshop on Statistical Machine Translation*.
- Keh-Jiann Chen and Shing-Huan Liu. 1992. Word identification for mandarin chinese sentences. In *Proceedings of COLING*.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *Proceedings of EMNLP*.
- John DeNero and Klaus Macherey. 2011. Model-based aligner combination using dual decomposition. In *Proceedings of ACL*.
- Thomas Emerson. 2005. The second international Chinese word segmentation bakeoff. In *Proceedings of the fourth SIGHAN workshop on Chinese language Processing*.
- Jianfeng Gao, Mu Li, and Chang-Ning Huang. 2003. Improved source-channel models for Chinese word segmentation. In *Proceedings of ACL*.
- Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of EMNLP*.
- Jonathan K. Kummerfeld, Daniel Tse, James R. Curran, and Dan Klein. 2013. An empirical examination of challenges in chinese parsing. In *Proceedings of ACL-Short*.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of 18th International Conference on Machine Learning (ICML)*.
- Dekang Lin. 2009. Combining language modeling and discriminative classification for word segmentation. In *Proceedings of the 10th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*.
- Alexander M. Rush and Michael Collins. 2012. A tutorial on dual decomposition and Lagrangian relaxation for inference in natural language processing. *JAIR*, 45:305–362.

- Alexander M. Rush, David Sontag, Michael Collins, and Tommi Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of EMNLP*.
- Yanxin Shi and Mengqiu Wang. 2007. A dual-layer crfs based joint decoding method for cascaded segmentation and labeling tasks. In *Proceedings of Joint Conferences on Artificial Intelligence (IJCAI)*.
- Richard Sproat and Thomas Emerson. 2003. The first international Chinese word segmentation bake-off. In *Proceedings of the second SIGHAN workshop on Chinese language Processing*.
- Xu Sun, Yaozhong Zhang, Takuya Matsuzaki, Yoshimasa Tsuruoka, and Jun'ichi Tsujii. 2009. A discriminative latent variable chinese segmenter with hybrid word/character information. In *Proceedings of HLT-NAACL*.
- Weiwei Sun. 2010. Word-based and character-based word segmentation models: Comparison and combination. In *Proceedings of COLING*.
- Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter for sighthan bake-off 2005. In *Proceedings of the fourth SIGHAN workshop on Chinese language Processing*.
- Xinhao Wang, Xiaojun Lin, Dianhai Yu, Hao Tian, and Xihong Wu. 2006. Chinese word segmentation with maximum entropy and n-gram language model. In *Proceedings of the fifth SIGHAN workshop on Chinese language Processing*.
- Kun Wang, Chengqing Zong, and Keh-Yih Su. 2010. A character-based joint model for chinese word segmentation. In *Proceedings of COLING*.
- Mengqiu Wang, Wanxiang Che, and Christopher D. Manning. 2013. Joint word alignment and bilingual named entity recognition using dual decomposition. In *Proceedings of ACL*.
- Nianwen Xue. 2003. Chinese word segmentation as character tagging. *International Journal of Computational Linguistics and Chinese Language Processing*, pages 29–48.
- Yue Zhang and Stephen Clark. 2007. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of ACL*.
- Ruiqiang Zhang, Genichiro Kikui, and Eiichiro Sumita. 2006. Subword-based tagging by conditional random fields for Chinese word segmentation. In *Proceedings of HLT-NAACL*.

Effective Document-Level Features for Chinese Patent Word Segmentation

Si Li

Chinese Language Processing Group
Brandeis University
Waltham, MA 02453, USA
lisi@brandeis.edu

Nianwen Xue

Chinese Language Processing Group
Brandeis University
Waltham, MA 02453, USA
xuen@brandeis.edu

Abstract

A patent is a property right for an invention granted by the government to the inventor. Patents often have a high concentration of scientific and technical terms that are rare in everyday language. However, some scientific and technical terms usually appear with high frequency only in one specific patent. In this paper, we propose a pragmatic approach to Chinese word segmentation on patents where we train a sequence labeling model based on a group of novel document-level features. Experiments show that the accuracy of our model reached 96.3% (F_1 score) on the development set and 95.0% on a held-out test set.

1 Introduction

It is well known that Chinese text does not come with natural word delimiters, and the first step for many Chinese language processing tasks is word segmentation, the automatic determination of word boundaries in Chinese text. Tremendous progress was made in this area in the last decade or so due to the availability of large-scale human segmented corpora coupled with better statistical modeling techniques. On the data side, there exist a few large-scale human annotated corpora based on established word segmentation standards, and these include the Chinese TreeBank (Xue et al., 2005), the Sinica Balanced Corpus (Chen et al., 1996), the PKU Peoples' Daily Corpus (Duan et al., 2003), and the LIVAC balanced corpus (T'sou et al., 1997). Another driver for the improvement in Chinese word segmentation accuracy comes from the evolution of statistical modeling techniques. Dictionaries used to play a central role in early heuristics-based word segmentation techniques (Chen and Liu, 1996; Sproat et al., 1996).

Modern word segmentation systems have moved away from dictionary-based approaches in favor of character tagging approaches. This allows the word segmentation problem to be modeled as a sequence labeling problem, and lends itself to discriminative sequence modeling techniques (Xue, 2003; Peng et al., 2004). With these better modeling techniques, state-of-the-art systems routinely report accuracy in the high 90%, and a few recent systems report accuracies of over 98% in F_1 score (Sun, 2011; Zeng et al., 2013b).

Chinese word segmentation is not a solved problem however and significant challenges remain. Advanced word segmentation systems perform very well in domains such as newswire where everyday language is used and there is a large amount of human annotated training data. There is often a rapid degradation in performance when systems trained on one domain (let us call it the *source* domain) are used to segment data in a different domain (let us call it the *target* domain). This problem is especially severe when the target domain is distant from the source domain. This is the problem we are facing when we perform word segmentation on Chinese patent data. The word segmentation accuracy on Chinese patents is very poor if the word segmentation model is trained on the Chinese TreeBank data, which consists of data sources from a variety of genres but no patents. To address this issue, we annotated a corpus of 142 patents which contain about 440K words according to the Chinese TreeBank standards. We trained a character-tagging based CRF model for word segmentation, and based on the writing style of patents, we propose a group of document-level features as well as a novel character part-of-speech feature (C_POS). Our results show these new features are effective and we are able to achieve an accuracy of 96.3% (F_1 score) on the development set and 95% (F_1 score) on the test set.

2 Method

We adopt the character-based sequence labeling approach, first proposed in (Xue, 2003), as our modeling technique for its simplicity and effectiveness. This approach treats each sentence as a sequence of characters and assigns to each character a label that indicates its position in the word. In this paper, we use the *BMES* tag set to indicate the character positions. The tag set has four labels that represent for possible positions a character can occupy within a word: *B* for beginning, *M* for middle, *E* for ending, and *S* for a single character as a word. After each character in a sentence is tagged with a *BMES* label, a sequence of words can be derived from this labeled character sequence.

We train a Conditional Random Field (CRF) (Lafferty et al., 2001) model for this sequence labeling. When extracting features to train a CRF model from a sequence of n characters $C_1C_2\dots C_{i-1}C_iC_{i+1}\dots C_n$, we extract features for each character C_i from a fixed window. We start with a set of core features extracted from the annotated corpus that have been shown to be effective in previous works and propose some new features for patent word segmentation. We describe each group of features in detail below.

2.1 Character features (CF)

When predicting the position of a character within a word, features based on its surrounding characters and their types have shown to be the most effective features for this task (Xue, 2003). There are some variations of these features depending on the window size in terms of the number of characters to examine, and here we adopt the feature templates used in (Ng and Low, 2004).

Character N-gram features The N-gram features are various combinations of the surrounding characters of the candidate character C_i . The 10 features we used are listed below:

- Character unigrams: C_k ($i - 3 < k < i + 3$)
- Character bigrams: C_kC_{k+1} ($i - 3 < k < i + 2$) and $C_{k-1}C_{k+1}$ ($k = i$)

Character type N-gram features We classify the characters in Chinese text into 4 types: Chinese characters or *hanzi*, English letters, numbers and others. T_i is the character type of C_i . The character type has been used in the previous works in various forms (Ng and Low, 2004; Jiang et al., 2009), and the 4 features we use are as follows:

- Character type unigrams: T_k ($k = i$)
- Character type bigrams: T_kT_{k+1} ($i - 2 < k < i + 1$) and $T_{k-1}T_{k+1}$ ($k = i$)

Starting with this baseline, we extract some new features to improve Chinese patent word segmentation accuracy.

2.2 POS of single-character words (C_POS)

Chinese words are composed of Chinese *hanzi*, and an overwhelming majority of these Chinese characters can be single-character words themselves in some context. In fact, most of the multi-character words are compounds that are 2-4 characters in length. The formation of these compound words is not random and abide by word formation rules that are similar to the formation of phrases (Xue, 2000; Packard, 2000). In fact, the Chinese TreeBank word segmentation guidelines (Xia, 2000) specify how words are segmented based on the part-of-speech (POS) of their component characters. We hypothesize that the POS tags of the single-character words would be useful information to help predict how they form the compound words, and these POS tags are more fine-grained information than the character type information described in the previous section, but are more robust and more generalizable than the characters themselves.

Since we do not have POS-tagged patent data, we extract this information from the Chinese TreeBank (CTB) 7.0, a 1.2-million-word out-of-domain dataset. We extract the POS tags for all the single-character words in the CTB. Some of the single-character words will have more than one POS tag. In this case, we select the POS tag with the highest frequency as the C_POS tag for this character. The result of this extraction process is a list of single-character Chinese words, each of which is assigned a single POS tag.

When extracting features for the target character C_i , if C_i is in this list, the POS tag of C_i is used as a feature for this target character.

2.3 Document-level features

A patent is a property right for an invention granted by the government to the inventor, and many of the patents have a high concentration of scientific and technical terms. From a machine learning perspective, these terms are hard to detect and segment because they are often "new words" that are not seen in everyday language. These technical

Algorithm 1 Longest n-gram sequence extraction.**Input:**Sentences $\{s_i\}$ in patent P_i ;**Output:**Longest n-gram sequence list for P_i ;

- 1: **For** each sentence s_i in P_i **do**:
n-gram sequence extraction
($2 \leq n \leq \text{length}(s_i)$);
 - 2: Count the frequency of each n-gram sequence;
 - 3: Delete the sequence if its frequency < 2 ;
 - 4: Delete sequence i if it is contained in a longer sequence j ;
 - 5: All the remaining sequences form a longest n-gram sequence list for P_i ;
 - 6: **return** Longest n-gram sequences list.
-

terminologies also tend to be very sparse, either because they are related to the latest invention that has not made into everyday language, or because our limited patent dataset cannot possibly cover all possible technical topics. However, these technical terms are also topical and they tend to have high relative frequency within a patent document even though they are sparse in the entire patent data set. We attempt to exploit this distribution property with some document-level features which are extracted based on each patent document.

Longest n-gram features (LNG) We propose a longest n-gram (LNG) feature as a document-level feature. Each patent document is treated as an independent unit and the candidate longest n-gram sequence lists for each patent are obtained as described in Algorithm 1.

For a given patent, the LNG feature value for the target character C_i 's LNG is set to 'S' if the bigram (C_i, C_{i+1}) are the first two characters of an n-gram sequence in this patent's longest n-gram sequence list. If (C_{i-1}, C_i) are the last two characters of an n-gram sequence in this patent's longest n-gram sequence list, the target character C_i 's LNG is set to 'F'. It is set to 'O' otherwise. If C_i can be labeled as both 'S' and 'F' at the same time, label 'T' will be given as the final label. For example, if ' α ' is the target character C_i in patent A and the sequence ' α -干扰素' is in patent A's longest n-gram sequence list. If the character next to ' α ' is '-', the value of the LNG feature is set to 'S'. If the next character is not '-', the value of the LNG feature is set to 'O'.

Algorithm 2 Pseudo KL divergence.**Input:**Sentences $\{s_i\}$ in patent P_i ;**Output:**Pseudo KL divergence values between different characters in P_i ;

- 1: **For** each sentence s_i in P_i **do**:
trigram sequences extraction;
- 2: Count the frequency of each trigram;
- 3: Delete the trigram if its frequency < 2 ;
- 4: **For** C_i in trigram $C_i C_{i+1} C_{i+2}$ **do** :

$$PKL(C_i, C_{i+1}) = p(C_i^1) \log \frac{p(C_i^1)}{p(C_{i+1}^2)} \quad (1)$$

$$PKL(C_i, C_{i+2}) = p(C_i^1) \log \frac{p(C_i^1)}{p(C_{i+2}^3)} \quad (2)$$

The superscripts $\{1,2,3\}$ indicate the character position in trigram sequences;

- 5: **return** $PKL(C_i, C_{i+1})$ and $PKL(C_i, C_{i+2})$ for the first character C_i in each trigram.
-

Pseudo Kullback-Leibler divergence (PKL)

The second document-level feature we propose is the Pseudo Kullback-Leibler divergence feature which is calculated following the form of the Kullback-Leibler divergence. The relative position information is very important for Chinese word segmentation as a sequence labeling task. Characters XY may constitute a meaningful word, but characters YX may not be. Therefore, if we want to determine whether character X and character Y can form a word, the relative position of these two characters should be considered. We adopt a pseudo KL divergence with the relative position information as a measure of the association strength between two adjacent characters X and Y . The pseudo KL divergence is an asymmetric measure. The PKL value between character X and character Y is described in Algorithm 2.

The PKL values are real numbers and are sparse. A common solution to sparsity reduction is binning. We rank the PKL values between two adjacent characters in each patent from low to high, and then divide all values into five bins. Each bin is assigned a unique ID and all PKL values in the same bin are replaced by this ID. This ID is then used as the PKL feature value for the target character C_i .

Pointwise Mutual information (PMI) Pointwise Mutual information has been widely used in previous work on Chinese word segmentation (Sun and Xu, 2011; Zhang et al., 2013b) and it is a measure of the mutual dependence of two strings and reflects the tendency of two strings appearing in one word. In previous work, PMI statistics are gathered on the entire data set, and here we gather PMI statistics for each patent in an attempt to capture character strings with high PMI in a particular patent. The procedure for calculating PMI is the same as that for computing pseudo KL divergence, but the functions (1) and (2) are replaced with the following functions:

$$PMI(C_i, C_{i+1}) = \log \frac{p(C_i^1, C_{i+1}^2)}{p(C_i^1)p(C_{i+1}^2)} \quad (3)$$

$$PMI(C_i, C_{i+2}) = \log \frac{p(C_i^1, C_{i+2}^3)}{p(C_i^1)p(C_{i+2}^3)} \quad (4)$$

For the target character C_i , we obtain the values for $PMI(C_i, C_{i+1})$ and $PMI(C_i, C_{i+2})$. In each patent document, we rank these values from high to low and divided them into five bins. Then the PMI feature values are represented by the bin IDs.

3 Experiments

3.1 Data preparation

We annotated 142 Chinese patents following the CTB word segmentation guidelines (Xia, 2000). Since the original guidelines are mainly designed to cover non-technical everyday language, many scientific and technical terms found in patents are not covered in the guidelines. We had to extend the CTB word segmentation guidelines to handle these new words. Deciding on how to segment these scientific and technical terms is a big challenge since these patents cover many different technical fields and without proper technical background, even a native speaker has difficulty in segmenting them properly. For difficult scientific and technical terms, we consult BaiduBaiké ("Baidu Encyclopedia")¹, which we use as a scientific and technical terminology dictionary during our annotation. There are still many words that do not appear in BaiduBaiké, and these include chemical names and formulas. These chemical names and formulas (e.g., “1-溴-3-氯丙烷/1-bromo-3-chloropropane”) are usually very

¹<http://baike.baidu.com/>

Table 1: Training, development and test data on Patent data

Data set	# of words	# of patent
Training	345336	113
Devel.	46196	14
Test	48351	15

long, and unlike everyday words, they often have numbers and punctuation marks in them. We decided not to try segmenting the internal structures of such chemical terms and treat them as single words, because without a technical background in chemistry, it is very hard to segment their internal structures consistently.

The annotated patent dataset covers many topics and they include chemistry, mechanics, medicine, etc. If we consider the words in our *annotated dataset* but not in CTB 7.0 data as *new words* (or out-of-vocabulary, OOV), the new words account for 18.3% of the patent corpus by token and 68.1% by type. This shows that there is a large number of words in the patent corpus that are not in the everyday language vocabulary. Table 1 presents the data split used in our experiments.

3.2 Main results

We use CRF++ (Kudo, 2013) to train our sequence labeling model. *Precision*, *recall*, F_1 score and R_{OOV} are used to evaluate our word segmentation methods, where R_{OOV} for our purposes means the recall of new words which do not appear in CTB 7.0 but in patent data.

Table 2 shows the segmentation results on the development and test sets with different feature templates and different training sets. The CTB training set includes the entire CTB 7.0, which has 1.2 million words. The model with the CF feature template is considered to be the baseline system. We conducted 4 groups of experiments based on the different datasets: (1) patent training set + patent development set; (2) patent training set + patent test set; (3) CTB training set + patent development set; (4) CTB training set + patent test set.

The results in Table 2 show that the models trained on the patent data outperform the models trained on the CTB data by a big margin on both the development and test set, even if the CTB training set is much bigger. That proves the importance of having a training set in the same do-

Table 2: Segmentation performance with different feature sets on different datasets.

Train set	Test set	Features	P	R	F_1	R_{OOV}
Patent train	Patent dev.	CF	95.34	95.28	95.32	90.02
		CF+C_POS	95.58	95.40	95.49	90.40
		CF+C_POS+LNG	96.32	96.00	96.15	91.22
		CF+C_POS+PKL	95.62	95.41	95.51	90.40
		CF+C_POS+PMI	95.65	95.40	95.53	89.94
		CF+C_POS+PMI+PKL	95.72	95.53	95.62	90.37
		CF+C_POS+LNG+PMI	96.42	96.09	96.26	91.66
		CF+C_POS+LNG+PMI+PKL	96.48	96.12	96.30	91.69
Patent train	Patent test	CF	93.98	94.49	94.23	85.19
		CF+C_POS+LNG+PKL+PMI	94.89	95.10	95.00	87.89
CTB train	Patent dev.	CF+C_POS+LNG+PKL+PMI	89.04	90.75	89.89	72.80
CTB train	Patent test	CF+C_POS+LNG+PKL+PMI	87.88	89.03	88.45	70.89

main. The results also show that adding the new features we proposed leads to consistent improvement across all experimental conditions, and that the LNG features are the most effective and bring about the largest improvement in accuracy.

4 Related work

Most of the previous work on Chinese word segmentation focused on newswire, and one widely adopted technique is character-based representation combined with sequential learning models (Xue, 2003; Low et al., 2005; Zhao et al., 2006; Sun and Xu, 2011; Zeng et al., 2013b; Zhang et al., 2013b; Wang and Kan, 2013). More recently, word-based models using perceptron learning techniques (Zhang and Clark, 2007) also produce very competitive results. There are also some recent successful attempts to combine character-based and word-based techniques (Sun, 2010; Zeng et al., 2013a).

As Chinese word segmentation has reached a very high accuracy in the newswire domain, the attention of the field has started to shift to other domains where there are few annotated resources and the problem is more challenging, such as work on the word segmentation of literature data (Liu and Zhang, 2012) and informal language genres (Wang and Kan, 2013; Zhang et al., 2013a). Patents are distinctly different from the above genres as they contain scientific and technical terms that require some special training to understand. There has been very little work in this area, and the only work that is devoted to Chinese word segmentation is (Guo et al., 2012), which reports

work on Chinese patent word segmentation with a fairly small test set without any annotated training data in the target domain. They reported an accuracy of 86.42% (F_1 score), but the results are incomparable with ours as their evaluation data is not available to us. We differ from their work in that we manually segmented a significant amount of data, and trained a model with document-level features designed to capture the characteristics of patent data.

5 Conclusion

In this paper, we presented an accurate character-based word segmentation model for Chinese patents. Our contributions are two-fold. Our first contribution is that we have annotated a significant amount of Chinese patent data and we plan to release this data once the copyright issues have been cleared. Our second contribution is that we designed document-level features to capture the distributional characteristics of the scientific and technical terms in patents. Experimental results showed that the document-level features we proposed are effective for patent word segmentation.

Acknowledgments

This paper is supported by the Intelligence Advanced Research Projects Activity (IARPA) via a contract NO. D11PC20154. All views expressed in this paper are those of the authors and do not necessarily represent the view of IARPA, DoI/NBC, or the U.S. Government.

References

- Keh-Jiann Chen and Shing-Huan Liu. 1996. Word Identification for Mandarin Chinese Sentences. In *Proceedings of COLING'92*, pages 101–107.
- 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.
- Huiming Duan, Xiaojing Bai, Baobao Chang, and Shiwen Yu. 2003. Chinese word segmentation at Peking University. In *Proceedings of the second SIGHAN workshop on Chinese language processing*, pages 152–155.
- Zhen Guo, Yujie Zhang, Chen Su, and Jinan Xu. 2012. Exploration of N-gram Features for the Domain Adaptation of Chinese Word Segmentation. In *Proceedings of Natural Language Processing and Chinese Computing Natural Language Processing and Chinese Computing*, pages 121–131.
- Wenbin Jiang, Liang Huang, and Qun Liu. 2009. Automatic Adaptation of Annotation Standards: Chinese Word Segmentation and POS Tagging - A Case Study. In *Proceedings of ACL'09*, pages 522–530.
- Taku Kudo. 2013. CRF++: Yet Another CRF toolkit.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML'01*, pages 282–289.
- Yang Liu and Yue Zhang. 2012. Unsupervised Domain Adaptation for Joint Segmentation and POS-Tagging. In *Proceedings of COLING'12*, pages 745–754.
- Jin Kiat Low, Hwee Tou Ng, and Wenyuan Guo. 2005. A Maximum Entropy Approach to Chinese Word Segmentation. In *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing*, pages 970–979.
- Hwee Tou Ng and Jin Kiat Low. 2004. Chinese Part-of-Speech Tagging: One-at-a-Time or All-at-Once? Word-Based or Character-Based? In *Proceedings of EMNLP'04*, pages 277–284.
- Jerome Packard. 2000. *The Morphology of Chinese: a cognitive and linguistic approach*. Cambridge University Press.
- Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese Segmentation and New Word Detection using Conditional Random Fields. In *Proceedings of COLING'04*.
- Richard Sproat, Chilin Shih, William Gale, and Nancy Chang. 1996. A Stochastic Finite-State Word-Segmentation Algorithm for Chinese. *Computational Linguistics*, 22(3):377–404.
- Weiwei Sun and Jia Xu. 2011. Enhancing Chinese Word Segmentation Using Unlabeled Data. In *Proceedings of EMNLP'11*, pages 970–979.
- Weiwei Sun. 2010. Word-based and character-based word segmentation models: Comparison and combination. In *Proceedings of ACL'10*, pages 1211–1219.
- Weiwei Sun. 2011. A Stacked Sub-Word Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. In *Proceedings of ACL'11*, pages 1385–1394.
- Benjamin K. T'sou, Hing-Lung Lin, Godfrey Liu, Terence Chan, Jerome Hu, Ching hai Chew, and John K.P. Tse. 1997. A Synchronous Chinese Language Corpus from Different Speech Communities: Construction and Application. *International Journal of Computational Linguistics and Chinese Language Processing*, 2(1):91–104.
- Aobo Wang and Min-Yen Kan. 2013. Mining Informal Language from Chinese Microtext: Joint Word Recognition and Segmentation. In *Proceedings of ACL'13*, pages 731–741.
- Fei Xia. 2000. The segmentation guidelines for the Penn Chinese Treebank (3.0).
- Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The Penn Chinese TreeBank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering*, 11(2):207–238.
- Nianwen Xue. 2000. *Defining and identifying words in Chinese*. Ph.D. thesis, University of Delaware.
- Nianwen Xue. 2003. Chinese Word Segmentation as Character Tagging. *International Journal of Computational Linguistics and Chinese Language Processing*, 8(1):29–48.
- Xiaodong Zeng, Derek F. Wong, Lidia S. Chao, and Isabel Trancoso. 2013a. Co-regularizing character-based and word-based models for semi-supervised Chinese word segmentation. In *Proceedings of ACL'13*, pages 171–176.
- Xiaodong Zeng, Derek F. Wong, Lidia S. Chao, and Isabel Trancoso. 2013b. Graph-based Semi-Supervised Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. In *Proceedings of ACL'13*, pages 770–779.
- Yue Zhang and Stephen Clark. 2007. Chinese Segmentation Using a Word-based Perceptron Algorithm. In *Proceedings of ACL'07*, pages 840–847.
- Longkai Zhang, Li Li, Zhengyan He, Houfeng Wang, and Ni Sun. 2013a. Improving Chinese Word Segmentation on Micro-blog Using Rich Punctuations. In *Proceedings of ACL'13*, pages 177–182.

Longkai Zhang, Houfeng Wang, Xu Sun, and Mairgup Mansur. 2013b. Exploring Representations from Unlabeled Data with Co-training for Chinese Word Segmentation. In *Proceedings of EMNLP'13*, pages 311–321.

Hai Zhao, Chang-Ning Huang, and Mu Li. 2006. An improved Chinese word segmentation system with conditional random field. In *Proceedings of the 5th SIGHAN Workshop on Chinese Language Processing*, pages 162–165.

Word Segmentation of Informal Arabic with Domain Adaptation

Will Monroe, Spence Green, and Christopher D. Manning

Computer Science Department, Stanford University

{wmonroe4, spenceg, manning}@stanford.edu

Abstract

Segmentation of clitics has been shown to improve accuracy on a variety of Arabic NLP tasks. However, state-of-the-art Arabic word segmenters are either limited to formal Modern Standard Arabic, performing poorly on Arabic text featuring dialectal vocabulary and grammar, or rely on linguistic knowledge that is hand-tuned for each dialect. We extend an existing MSA segmenter with a simple domain adaptation technique and new features in order to segment informal and dialectal Arabic text. Experiments show that our system outperforms existing systems on newswire, broadcast news and Egyptian dialect, improving segmentation F_1 score on a recently released Egyptian Arabic corpus to 95.1%, compared to 90.8% for another segmenter designed specifically for Egyptian Arabic.

1 Introduction

Segmentation of words, clitics, and affixes is essential for a number of natural language processing (NLP) applications, including machine translation, parsing, and speech recognition (Chang et al., 2008; Tsarfaty, 2006; Kurimo et al., 2006). Segmentation is a common practice in Arabic NLP due to the language’s morphological richness. Specifically, clitic separation has been shown to improve performance on Arabic parsing (Green and Manning, 2010) and Arabic-English machine translation (Habash and Sadat, 2006). However, the variety of Arabic dialects presents challenges in Arabic NLP. Dialectal Arabic contains non-standard orthography, vocabulary, morphology, and syntax. Tools that depend on corpora or grammatical properties that only consider formal Modern Standard Arabic (MSA) do not perform well when confronted with these differences. The creation of annotated corpora in dialectal Arabic (Maamouri et al., 2006) has promoted

the development of new systems that support dialectal Arabic, but these systems tend to be tailored to specific dialects and require separate efforts for Egyptian Arabic, Levantine Arabic, Maghrebi Arabic, etc.

We present a single clitic segmentation model that is accurate on both MSA and informal Arabic. The model is an extension of the character-level conditional random field (CRF) model of Green and DeNero (2012). Our work goes beyond theirs in three aspects. First, we handle two Arabic orthographic normalization rules that commonly require rewriting of tokens after segmentation. Second, we add new features that improve segmentation accuracy. Third, we show that dialectal data can be handled in the framework of *domain adaptation*. Specifically, we show that even simple feature space augmentation (Daumé, 2007) yields significant improvements in task accuracy.

We compare our work to the original Green and DeNero model and two other Arabic segmentation systems: the MADA+TOKAN toolkit v. 3.1 (Habash et al., 2009) and its Egyptian dialect variant, MADA-ARZ v. 0.4 (Habash et al., 2013). We demonstrate that our system achieves better performance across the board, beating all three systems on MSA newswire, informal broadcast news, and Egyptian dialect. Our segmenter achieves a 95.1% F_1 segmentation score evaluated against a gold standard on Egyptian dialect data, compared to 90.8% for MADA-ARZ and 92.9% for Green and DeNero. In addition, our model decodes input an order of magnitude faster than either version of MADA. Like the Green and DeNero system, but unlike MADA and MADA-ARZ, our system does not rely on a morphological analyzer, and can be applied directly to any dialect for which segmented training data is available. The source code is available in the latest public release of the Stanford Word Segmenter (<http://nlp.stanford.edu/software/segmenter.shtml>).

2 Arabic Word Segmentation Model

A CRF model (Lafferty et al., 2001) defines a distribution $p(\mathbf{Y}|\mathbf{X}; \theta)$, where $\mathbf{X} = \{x_1, \dots, x_N\}$ is the observed input sequence and $\mathbf{Y} = \{y_1, \dots, y_N\}$ is the sequence of labels we seek to predict. Green and DeNero use a linear-chain model with \mathbf{X} as the sequence of input *characters*, and \mathbf{Y}^* chosen according to the decision rule

$$\mathbf{Y}^* = \arg \max_{\mathbf{Y}} \sum_{i=1}^N \theta^\top \phi(\mathbf{X}, y_i, \dots, y_{i-3}, i).$$

where ϕ is the feature map defined in Section 2.1. Their model classifies each y_i as one of I (continuation of a segment), O (whitespace outside any segment), B (beginning of a segment), or F (pre-grouped foreign characters).

Our segmenter expands this label space in order to handle two Arabic-specific orthographic rules. In our model, each y_i can take on one of the six values $\{\text{I, O, B, F, REWAL, REWTA}\}$:

- REWAL indicates that the current character, which is always the Arabic letter ل l , starts a new segment and should additionally be transformed into the definite article ال al - when segmented. This type of transformation occurs after the prefix لي li - “to”.
- REWTA indicates that the current character, which is always the Arabic letter ت t , is a continuation but should be transformed into the letter ه h when segmented. Arabic orthography rules restrict the occurrence of ه h to the word-final position, writing it instead as ت t whenever it is followed by a suffix.

2.1 Features

The model of Green and DeNero is a third-order (i.e., 4-gram) Markov CRF, employing the following indicator features:

- a five-character window around the current character: for each $-2 \leq \delta \leq 2$ and $1 \leq i \leq N$, the triple $(x_{i+\delta}, \delta, y_i)$
- n -grams consisting of the current character and up to three preceding characters: for each $2 \leq n \leq 4$ and $n \leq i \leq N$, the character-sequence/label-sequence pair $(x_{i-n+1} \dots x_i, y_{i-n+1} \dots y_i)$
- whether the current character is punctuation

- whether the current character is a digit
- the Unicode block of the current character
- the Unicode character class of the current character

In addition to these, we include two other types of features motivated by specific errors the original system made on Egyptian dialect development data:

- Word length and position within a word: for each $1 \leq i \leq N$, the pairs (ℓ, y_i) , (a, y_i) , and (b, y_i) , where ℓ , a , and b are the total length of the word containing x_i , the number of characters after x_i in the word, and the number of characters before x_i in the word, respectively. Some incorrect segmentations produced by the original system could be ruled out with the knowledge of these statistics.
- First and last two characters of the current word, separately influencing the first two labels and the last two labels: for each word consisting of characters $x_s \dots x_t$, the tuples $(x_s x_{s+1}, x_{t-1} x_t, y_s y_{s+1}$, “begin”) and $(x_s x_{s+1}, x_{t-1} x_t, y_{t-1} y_t$, “end”). This set of features addresses a particular dialectal Arabic construction, the negation ما $mā$ - + [verb] + ش $-sh$, which requires a matching prefix and suffix to be segmented simultaneously. This feature set also allows the model to take into account other interactions between the beginning and end of a word, particularly those involving the definite article ال al -.

A notable property of this feature set is that it remains highly dialect-agnostic, even though our additional features were chosen in response to errors made on text in Egyptian dialect. In particular, it does not depend on the existence of a dialect-specific lexicon or morphological analyzer. As a result, we expect this model to perform similarly well when applied to other Arabic dialects.

2.2 Domain adaptation

In this work, we train our model to segment Arabic text drawn from three domains: newswire, which consists of formal text in MSA; broadcast news, which contains scripted, formal MSA as well as extemporaneous dialogue in a mix of MSA and dialect; and discussion forum posts written primarily in Egyptian dialect.

Model	Training Data	F ₁ (%)			TEDEval (%)		
		ATB	BN	ARZ	ATB	BN	ARZ
GD	ATB	97.60	94.87	79.92	98.22	96.81	87.30
GD	+BN+ARZ	97.28	96.37	92.90	98.05	97.45	95.01
+Rew	ATB	97.55	94.95	79.95	98.72	97.45	87.54
+Rew	+BN	97.58	96.60	82.94	98.75	98.18	89.43
+Rew	+BN+ARZ	97.30	96.09	92.64	98.59	97.91	95.03
+Rew+DA	+BN+ARZ	97.71	96.57	93.87	98.79	98.14	95.86
+Rew+DA+Feat	+BN+ARZ	98.36	97.35	95.06	99.14	98.57	96.67

Table 1: Development set results. **GD** is the model of Green and DeNero (2012). **Rew** is support for orthographic rewrites with the **REWAL** and **REWTA** labels. The fifth row shows the strongest baseline, which is the GD+Rew model trained on the concatenated training sets from all three treebanks. **DA** is domain adaptation via feature space augmentation. **Feat** adds the additional feature templates described in section 2.1. **ATB** is the newswire ATB; **BN** is the Broadcast News treebank; **ARZ** is the Egyptian treebank. Best results (**bold**) are statistically significant ($p < 0.001$) relative to the strongest baseline.

The approach to domain adaptation we use is that of *feature space augmentation* (Daumé, 2007). Each indicator feature from the model described in Section 2.1 is replaced by $N + 1$ features in the augmented model, where N is the number of domains from which the data is drawn (here, $N = 3$). These $N + 1$ features consist of the original feature and N “domain-specific” features, one for each of the N domains, each of which is active only when both the original feature is present and the current text comes from its assigned domain.

3 Experiments

We train and evaluate on three corpora: parts 1–3 of the newswire Arabic Treebank (ATB),¹ the Broadcast News Arabic Treebank (BN),² and parts 1–8 of the BOLT Phase 1 Egyptian Arabic Treebank (ARZ).³ These correspond respectively to the domains in section 2.2. We target the segmentation scheme used by these corpora (leaving morphological affixes and the definite article attached). For the ATB, we use the same split as Chiang et al. (2006). For each of the other two corpora, we split the data into 80% training, 10% development, and 10% test in chronological order by document.⁴ We train the Green and DeNero model and our improvements using L-BFGS with L_2 regularization.

¹LDC2010T13, LDC2011T09, LDC2010T08

²LDC2012T07

³LDC2012E{93,98,89,99,107,125}, LDC2013E{12,21}

⁴These splits are publicly available at <http://nlp.stanford.edu/software/parser-arabic-data-splits.shtml>.

3.1 Evaluation metrics

We use two evaluation metrics in our experiments. The first is an F_1 precision-recall measure, ignoring orthographic rewrites. F_1 scores provide a more informative assessment of performance than word-level or character-level accuracy scores, as over 80% of tokens in the development sets consist of only one segment, with an average of one segmentation every 4.7 tokens (or one every 20.4 characters).

The second metric we use is the TEDEval metric (Tsarfaty et al., 2012). TEDEval was developed to evaluate joint segmentation and parsing⁵ in Hebrew, which requires a greater variety of orthographic rewrites than those possible in Arabic. Its edit distance-based scoring algorithm is robust enough to handle the rewrites produced by both MADA and our segmenter.

We measure the statistical significance of differences in these metrics with an approximate randomization test (Yeh, 2000; Padó, 2006), with $R = 10,000$ samples.

3.2 Results

Table 1 contains results on the development set for the model of Green and DeNero and our improvements. Using domain adaptation alone helps performance on two of the three datasets (with a statistically insignificant decrease on broadcast news), and that our additional features further improve

⁵In order to evaluate segmentation in isolation, we convert each segmented sentence from both the model output and the gold standard to a flat tree with all segments descending directly from the root.

	F ₁ (%)			TEDEval (%)		
	ATB	BN	ARZ	ATB	BN	ARZ
MADA	97.36	94.54	78.35	97.62	96.96	86.78
MADA-ARZ	92.83	91.89	90.76	91.26	91.10	90.39
GD+Rew+DA+Feat	98.30	97.17	95.13	99.10	98.42	96.75

Table 2: Test set results. Our final model (last row) is trained on all available data (ATB+BN+ARZ). Best results (**bold**) are statistically significant ($p < 0.001$) relative to each MADA version.

	ATB	BN	ARZ
MADA	705.6 ± 5.1	472.0 ± 0.8	767.8 ± 1.9
MADA-ARZ	784.7 ± 1.6	492.1 ± 4.2	779.0 ± 2.7
GD+Rew+DA+Feat	90.0 ± 1.0	59.5 ± 0.3	72.7 ± 0.2

Table 3: Wallclock time (in seconds) for MADA, MADA-ARZ, and our model for decoding each of the three development datasets. Means and standard deviations were computed for 10 independent runs. MADA and MADA-ARZ are single-threaded. Our segmenter supports multithreaded execution, but the times reported here are for single-threaded runs.

segmentation on all datasets. Table 2 shows the segmentation scores our model achieves when evaluated on the three test sets, as well as the results for MADA and MADA-ARZ. Our segmenter achieves higher scores than MADA and MADA-ARZ on all datasets under both evaluation metrics. In addition, our segmenter is faster than MADA. Table 3 compares the running times of the three systems. Our segmenter achieves a 7x or more speedup over MADA and MADA-ARZ on all datasets.

4 Error Analysis

We sampled 100 errors randomly from all errors made by our final model (trained on all three datasets with domain adaptation and additional features) on the ARZ development set; see Table 4. These errors fall into three general categories:

- typographical errors and annotation inconsistencies in the gold data;
- errors that can be fixed with a fuller analysis of just the problematic token, and therefore represent a deficiency in the feature set; and
- errors that would require additional context or sophisticated semantic awareness to fix.

4.1 Typographical errors and annotation inconsistencies

Of the 100 errors we sampled, 33 are due to typographical errors or inconsistencies in the gold data.

We classify 7 as typos and 26 as annotation inconsistencies, although the distinction between the two is murky: typos are intentionally preserved in the treebank data, but segmentation of typos varies depending on how well they can be reconciled with standard Arabic orthography. Four of the seven typos are the result of a missing space, such as:

- يسهر بالليالي *yashar-bi-'l-layālī* “stays awake at night” (يسهر *yashar* + بي *bi-* + الليالي *al-layālī*)
- عملتأن *amilatnā-an* “madeus” (عملت *amilat* + أنا *-nā* + أن *an*)

The first example is segmented in the Egyptian treebank but is left unsegmented by our system; the second is left as a single token in the treebank but is split into the above three segments by our system.

Of the annotation inconsistencies that do not involve typographical errors, a handful are segmentation mistakes; however, in the majority of these cases, the annotator chose not to segment a word for justifiable but arbitrary reasons. In particular, a few colloquial “filler” expressions are sometimes not segmented, despite being compound Arabic words that are segmented elsewhere in the data. These include ربنا *rabbīnā* “[our] Lord” (oath); عندما *indamā* “when”/“while”; and خليك *khallīk* “keep”/“stay”. Also, tokens containing foreign words are sometimes not segmented, despite carrying Arabic affixes. An example of this is ومستر

Category	# of errors
Abnormal gold data	33
Typographical error	7
Annotation inconsistency	26
Need full-token features	36
Need more context	31
ولا <i>wlā</i>	5
نا <i>-nā</i> : verb/pron	7
ي <i>-y</i> : <i>nisba</i> /pron	4
other	15

Table 4: Counts of error categories (out of 100 randomly sampled ARZ development set errors).

wamistur “and *Mister* [English]”, which could be segmented as *wa-* + *مستر* *mistur*.

4.2 Features too local

In 36 of the 100 sampled errors, we conjecture that the presence of the error indicates a shortcoming of the feature set, resulting in segmentations that make sense locally but are not plausible given the full token. Two examples of these are:

- *wafīṭarīqah* “and in the way” segmented as *wa-* + *فطريقة* *fīṭarīqah* (correct analysis is *wa-* + *ف* *fī-* + *طريقة* *ṭarīqah*). *فطر* *fṭr* “break”/“breakfast” is a common Arabic root, but the presence of *ق* *q* should indicate that *فطر* *fṭr* is not the root in this case.
- *walāyuhimhum* “and it’s not important to them” segmented as *wa-* + *لا* *li-* + *ايهم* *-ayuhimm* + *هم* *-hum* (correct analysis is *wa-* + *لا* *lā* + *يهم* *yuhimm* + *هم* *-hum*). The 4-character window *لايه* *lāyh* occurs commonly with a segment boundary after the *ل* *l*, but the segment *ايهم* *-ayuhimm* is not a well-formed Arabic word.

4.3 Context-sensitive segmentations and multiple word senses

In the remaining 31 of 100 errors, external context is needed. In many of these, it is not clear how to address the error without sophisticated semantic reasoning about the surrounding sentence.

One token accounts for five of these errors: *ولا* *wlā*, which in Egyptian dialect can be analyzed as *wa-* + *لا* *lā* “and [do/does] not” or as *ولا* *wallā*

“or”. In a few cases, either is syntactically correct, and the meaning must be inferred from context.

Two other ambiguities are a frequent cause of error and seem to require sophisticated disambiguation. The first is *نا* *-nā*, which is both a first person plural object pronoun and a first person plural past tense ending. The former is segmented, while the latter is not. An example of this is the pair *علمنا* *ilmunā* “our knowledge” (*علم* *ilmu* + *نا* *-nā*) versus *علمنا* *alimnā* “we knew” (one segment). The other is *ي* *-y*, which is both a first person singular possessive pronoun and the *nisba* adjective ending (which turns a noun into an adjective meaning “of or related to”); only the former is segmented. One example of this distinction that appeared in the development set is the pair *موضوعي* *mawḍūṭī* “my topic” (*موضوع* *mawḍūṭ* + *ي* *-y*) versus *موضوعي* *mawḍūṭīy* “topical”, “objective”.

5 Conclusion

In this paper we demonstrate substantial gains on Arabic clitic segmentation for both formal and dialectal text using a single model with dialect-independent features and a simple domain adaptation strategy. We present a new Arabic segmenter which performs better than tools employing sophisticated linguistic analysis, while also giving impressive speed improvements. We evaluated our segmenter on broadcast news and Egyptian Arabic due to the current availability of annotated data in these domains. However, as data for other Arabic dialects and genres becomes available, we expect that the model’s simplicity and the domain adaptation method we use will allow the system to be applied to these dialects with minimal effort and without a loss of performance in the original domains.

Acknowledgments

We thank the three anonymous reviewers, and Reut Tsarfaty for valuable correspondence regarding TEDEval. The second author is supported by a National Science Foundation Graduate Research Fellowship. This work was supported by the Defense Advanced Research Projects Agency (DARPA) Broad Operational Language Translation (BOLT) program through IBM. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA or the US government.

References

- Pi-Chuan Chang, Michel Galley, and Christopher D. Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *WMT*.
- David Chiang, Mona T. Diab, Nizar Habash, Owen Rambow, and Safiullah Shareef. 2006. Parsing Arabic dialects. In *EACL*.
- Hal Daumé, III. 2007. Frustratingly easy domain adaptation. In *ACL*.
- Spence Green and John DeNero. 2012. A class-based agreement model for generating accurately inflected translations. In *ACL*.
- Spence Green and Christopher D. Manning. 2010. Better Arabic parsing: Baselines, evaluations, and analysis. In *COLING*.
- Nizar Habash and Fatiha Sadat. 2006. Arabic preprocessing schemes for statistical machine translation. In *NAACL, Short Papers*.
- Nizar Habash, Owen Rambow, and Ryan Roth. 2009. MADA+TOKAN: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization. In *MEDAR*.
- Nizar Habash, Ryan Roth, Owen Rambow, Ramy Eskander, and Nadi Tomeh. 2013. Morphological analysis and disambiguation for dialectal Arabic. In *HLT-NAACL*.
- Mikko Kurimo, Antti Puurula, Ebru Arisoy, Vesa Sivola, Teemu Hirsimäki, Janne Pykkönen, Tanel Alumäe, and Murat Saraclar. 2006. Unlimited vocabulary speech recognition for agglutinative languages. In *HLT-NAACL*.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.
- Mohamed Maamouri, Ann Bies, Tim Buckwalter, Mona Diab, Nizar Habash, Owen Rambow, and Dalila Tabessi. 2006. Developing and using a pilot dialectal Arabic treebank. In *LREC*.
- Sebastian Padó, 2006. *User's guide to sigf: Significance testing by approximate randomisation*. <http://www.nlpado.de/~sebastian/software/sigf.shtml>.
- Reut Tsarfaty, Joakim Nivre, and Evelina Andersson. 2012. Joint evaluation of morphological segmentation and syntactic parsing. In *ACL, Short Papers*.
- Reut Tsarfaty. 2006. Integrated morphological and syntactic disambiguation for Modern Hebrew. In *COLING-ACL*.
- Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. In *COLING*.

Resolving Lexical Ambiguity in Tensor Regression Models of Meaning

Dimitri Kartsaklis

University of Oxford

Department of

Computer Science

Wolfson Bldg, Parks Road

Oxford, OX1 3QD, UK

dimitri.kartsaklis@cs.ox.ac.uk

Nal Kalchbrenner

University of Oxford

Department of

Computer Science

Wolfson Bldg, Parks Road

Oxford, OX1 3QD, UK

nkalch@cs.ox.ac.uk

Mehrnoosh Sadrzadeh

Queen Mary Univ. of London

School of Electronic Engineering

and Computer Science

Mile End Road

London, E1 4NS, UK

mehrnoosh.sadrzadeh@qmul.ac.uk

Abstract

This paper provides a method for improving tensor-based compositional distributional models of meaning by the addition of an explicit disambiguation step prior to composition. In contrast with previous research where this hypothesis has been successfully tested against relatively simple compositional models, in our work we use a robust model trained with linear regression. The results we get in two experiments show the superiority of the prior disambiguation method and suggest that the effectiveness of this approach is model-independent.

1 Introduction

The provision of compositionality in distributional models of meaning, where a word is represented as a vector of co-occurrence counts with every other word in the vocabulary, offers a solution to the fact that no text corpus, regardless of its size, is capable of providing reliable co-occurrence statistics for anything but very short text constituents. By *composing* the vectors for the words within a sentence, we are still able to create a vectorial representation for that sentence that is very useful in a variety of natural language processing tasks, such as paraphrase detection, sentiment analysis or machine translation. Hence, given a sentence $w_1 w_2 \dots w_n$, a compositional distributional model provides a function f such that:

$$\vec{s} = f(\vec{w}_1, \vec{w}_2, \dots, \vec{w}_n) \quad (1)$$

where \vec{w}_i is the distributional vector of the i th word in the sentence and \vec{s} the resulting composite sentential vector.

An interesting question that has attracted the attention of researchers lately refers to the way in which these models affect ambiguous words; in other words, given a sentence such as “a man was waiting by the bank”, we are interested to know to what extent a composite vector can appropriately

reflect the intended use of word ‘bank’ in that context, and how such a vector would differ, for example, from the vector of the sentence “a fisherman was waiting by the bank”.

Recent experimental evidence (Reddy et al., 2011; Kartsaklis et al., 2013; Kartsaklis and Sadrzadeh, 2013) suggests that for a number of compositional models the introduction of a disambiguation step *prior* to the actual compositional process results in better composite representations. In other words, the suggestion is that Eq. 1 should be replaced by:

$$\vec{s} = f(\phi(\vec{w}_1), \phi(\vec{w}_2), \dots, \phi(\vec{w}_n)) \quad (2)$$

where the purpose of function ϕ is to return a disambiguated version of each word vector given the rest of the context (e.g. all the other words in the sentence). The composition operation, whatever that could be, is then applied on these unambiguous representations of the words, instead of the original distributional vectors.

Until now this idea has been verified on relatively simple compositional functions, usually involving some form of element-wise operation between the word vectors, such as addition or multiplication. An exception to this is the work of Kartsaklis and Sadrzadeh (2013), who apply Eq. 2 on *partial* tensor-based compositional models. In a tensor-based model, relational words such as verbs and adjectives are represented by multi-linear maps; composition takes place as the application of those maps on vectors representing the arguments (usually nouns). What makes the models of the above work ‘partial’ is that the authors used simplified versions of the linear maps, projected onto spaces of order lower than that required by the theoretical framework. As a result, a certain amount of transformational power was traded off for efficiency.

A potential explanation then for the effectiveness of the proposed prior disambiguation method can be sought on the limitations imposed by the compositional models under test. After all, the idea of having disambiguation emerge as a direct

consequence of the compositional process, without the introduction of any explicit step, seems more natural and closer to the way the human mind resolves lexical ambiguities.

The purpose of this paper is to investigate the hypothesis whether prior disambiguation is important in a pure tensor-based compositional model, where no simplifying assumptions have been made. We create such a model by using linear regression, and we explain how an explicit disambiguation step can be introduced to this model prior to composition. We then proceed by comparing the composite vectors produced by this approach with those produced by the model alone in a number of experiments. The results show a clear superiority of the priorly disambiguated models following Eq. 2, confirming previous research and suggesting that the reasons behind the success of this approach are more fundamental than the form of the compositional function.

2 Composition in distributional models

Compositional distributional models of meaning vary in sophistication, from simple element-wise operations between vectors such as addition and multiplication (Mitchell and Lapata, 2008) to deep learning techniques based on neural networks (Socher et al., 2011; Socher et al., 2012; Kalchbrenner and Blunsom, 2013a). *Tensor-based models*, formalized by Coecke et al. (2010), comprise a third class of models lying somewhere in between these two extremes. Under this setting relational words such as verbs and adjectives are represented by multi-linear maps (tensors of various orders) acting on a number of arguments. An adjective for example is a linear map $f : N \rightarrow N$ (where N is our basic vector space for nouns), which takes as input a noun and returns a modified version of it. Since every map of this sort can be represented by a matrix living in the tensor product space $N \otimes N$, we now see that the meaning of a phrase such as ‘red car’ is given by $\overrightarrow{red} \times \overrightarrow{car}$, where \overrightarrow{red} is an adjective matrix and \times indicates matrix multiplication. The same concept applies for functions of higher order, such as a transitive verb (a function of two arguments, so a tensor of order 3). For these cases, matrix multiplication generalizes to the more generic notion of *tensor contraction*. The meaning of a sentence such as ‘kids play games’ is computed as:

$$\overrightarrow{kids}^T \times \overrightarrow{play} \times \overrightarrow{games} \quad (3)$$

where \overrightarrow{play} here is an order-3 tensor (a ‘‘cube’’) and \times now represents tensor contraction. A con-

cise introduction to compositional distributional models can be found in (Kartsaklis, 2014).

3 Disambiguation and composition

The idea of separating disambiguation from composition first appears in a work of Reddy et al. (2011), where the authors show that the introduction of an explicit disambiguation step prior to simple element-wise composition is beneficial for noun-noun compounds. Subsequent work by Kartsaklis et al. (2013) reports very similar findings for verb-object structures, again on additive and multiplicative models. Finally, in (Kartsaklis and Sadrzadeh, 2013) these experiments were extended to include tensor-based models following the categorical framework of Coecke et al. (2010), where again all ‘‘unambiguous’’ models present superior performance compared to their ‘‘ambiguous’’ versions.

However, in this last work one of the dimensions of the tensors was kept empty (filled in with zeros). This simplified the calculations but also weakened the effectiveness of the multi-linear maps. If, for example, instead of using an order-3 tensor for a transitive verb, one uses some of the matrix instantiations of Kartsaklis and Sadrzadeh, Eq. 3 is reduced to one of the following forms:

$$\overrightarrow{play} \odot (\overrightarrow{kids} \otimes \overrightarrow{games}), \quad \overrightarrow{kids} \odot (\overrightarrow{play} \times \overrightarrow{games}) \quad (4)$$

$$(\overrightarrow{kids}^T \times \overrightarrow{play}) \odot \overrightarrow{games}$$

where symbol \odot denotes element-wise multiplication and \overrightarrow{play} is a matrix. Here, the model does not fully exploit the space provided by the theoretical framework (i.e. an order-3 tensor), which has two disadvantages: firstly, we lose space that could hold valuable information about the verb in this case and relational words in general; secondly, the generally non-commutative tensor contraction operation is now partly relying on element-wise multiplication, which is commutative, thus forgets (part of the) order of composition.

In the next section we will see how to apply linear regression in order to create full tensors for verbs and use them for a compositional model that avoids these pitfalls.

4 Creating tensors for verbs

The essence of any tensor-based compositional model is the way we choose to create our sentence-producing maps, i.e. the verbs. In this paper we adopt a method proposed by Baroni and Zamparelli (2010) for building adjective matrices, which can be generally applied to any relational word.

In order to create a matrix for, say, the intransitive verb ‘play’, we first collect all instances of the verb occurring with some subject in the training corpus, and then we create non-compositional holistic vectors for these elementary sentences following exactly the same methodology as if they were words. We now have a dataset with instances of the form $\langle \overrightarrow{subj_i}, \overrightarrow{subj_i \text{ play}} \rangle$ (e.g. the vector of ‘kids’ paired with the holistic vector of ‘kids play’, and so on), that can be used to train a linear regression model in order to produce an appropriate matrix for verb ‘play’. The premise of a model like this is that the multiplication of the verb matrix with the vector of a new subject will produce a result that approximates the distributional behaviour of all these elementary two-word exemplars used in training.

We present examples and experiments based on this method, constructing ambiguous and disambiguated tensors of order 2 (that is, matrices) for verbs taking one argument. In principle, our method is directly applicable to tensors of higher order, following a multi-step process similar to that of Grefenstette et al. (2013) who create order-3 tensors for transitive verbs using similar means. Instead of using subject-verb constructs as above we concentrate on elementary verb phrases of the form *verb-object* (e.g. ‘play football’, ‘admit student’), since in general objects comprise stronger contexts for disambiguating the usage of a verb.

5 Experimental setting

Our basic vector space is trained from the ukWaC corpus (Ferraresi et al., 2008), originally using as a basis the 2,000 content words with the highest frequency (but excluding a list of stop words as well as the 50 most frequent content words since they exhibit low information content). We created vectors for all content words with at least 100 occurrences in the corpus. As context we considered a 5-word window from either side of the target word, while as our weighting scheme we used local mutual information (i.e. point-wise mutual information multiplied by raw counts). This initial semantic space achieved a score of 0.77 Spearman’s ρ (and 0.71 Pearson’s r) on the well-known benchmark dataset of Rubenstein and Goodenough (1965). In order to reduce the time of regression training, our vector space was normalized and projected onto a 300-dimensional space using singular value decomposition (SVD). The performance of the reduced space on the R&G dataset was again very satisfying, specifically 0.73 Spearman’s ρ and 0.72 Pearson’s r .

In order to create the vector space of the holistic verb phrase vectors, we first collected all instances where a verb participating in the experiments appeared at least 100 times in a verb-object relationship with some noun in the corpus. As context of a verb phrase we considered any content word that falls into a 5-word window from either side of the verb or the object. For the 68 verbs participating in our experiments, this procedure resulted in 22k verb phrases, a vector space that again was projected into 300 dimensions using SVD.

Linear regression For each verb we use simple linear regression with gradient descent directly applied on matrices \mathbf{X} and \mathbf{Y} , where the rows of \mathbf{X} correspond to vectors of the nouns that appear as objects for the given verb and the rows of \mathbf{Y} to the holistic vectors of the corresponding verb phrases. Our objective function then becomes:

$$\hat{\mathbf{W}} = \arg \min_{\mathbf{W}} \frac{1}{2m} \left(\|\mathbf{W}\mathbf{X}^T - \mathbf{Y}^T\|^2 + \lambda \|\mathbf{W}\|^2 \right) \quad (5)$$

where m is the number of training examples and λ a regularization parameter. The matrix \mathbf{W} is used as the tensor for the specific verb.

6 Supervised disambiguation

In our first experiment we test the effectiveness of a prior disambiguation step for a tensor-based model in a “sandbox” using supervised learning. The goal is to create composite vectors for a number of elementary verb phrases of the form *verb-object* with and without an explicit disambiguation step, and evaluate which model approximates better the holistic vectors of these verb phrases.

The verb phrases of our dataset are based on the 5 ambiguous verbs of Table 1. Each verb has been combined with two different sets of nouns that appear in a verb-object relationship with that verb in the corpus (a total of 343 verb phrases). The nouns of each set have been manually selected in order to explicitly represent a different meaning of the verb. As an example, in the verb ‘play’ we impose the two distinct meanings of using a musical instrument and participating in a sport; so the first

Verb	Meaning 1	Meaning 2
break	violate (56)	break (22)
catch	capture (28)	be on time (21)
play	musical instrument (47)	sports (29)
admit	permit to enter (12)	acknowledge (25)
draw	attract (64)	sketch (39)

Table 1: Ambiguous verbs for the supervised task. The numbers in parentheses refer to the collected training examples for each case.

set of objects contains nouns such as ‘oboe’, ‘piano’, ‘guitar’, and so on, while in the second set we see nouns such as ‘football’, ‘baseball’ etc.

In more detail, the creation of the dataset was done in the following way: First, all verb entries with more than one definition in the Oxford Junior Dictionary (Sansome et al., 2000) were collected into a list. Next, a linguist (native speaker of English) annotated the semantic difference between the definitions of each verb in a scale from 1 (similar) to 5 (distinct). Only verbs with definitions exhibiting completely distinct meanings (marked with 5) were kept for the next step. For each one of these verbs, a list was constructed with all the nouns that appear at least 50 times under a verb-object relationship in the corpus with the specific verb. Then, each object in the list was manually annotated as *exclusively* belonging to one of the two senses; so, an object could be selected only if it was related to a single sense, but not both. For example, ‘attention’ was a valid object for the *attract* sense of verb ‘draw’, since it is unrelated to the *sketch* sense of that verb. On the other hand, ‘car’ is not an appropriate object for either sense of ‘draw’, since it could actually appear under both of them in different contexts. The verbs of Table 1 were the ones with the highest numbers of exemplars per sense, creating a dataset of significant size for the intended task (each holistic vector is compared with 343 composite vectors).

We proceed as follows: We apply linear regression in order to train verb matrices using jointly the object sets for both meanings of each verb, as well as separately—so in this latter case we get two matrices for each verb, one for each sense. For each verb phrase, we create a composite vector by matrix-multiplying the verb matrix with the vector of the specific object. Then we use 4-fold cross validation to evaluate which version of composite vectors (the one created by the ambiguous tensors or the one created by the unambiguous ones) approximates better the holistic vectors of the verb phrases in our test set. This is done by comparing each holistic vector with all the composite ones, and then evaluating the rank of the correct composite vector within the list of results.

In order to get a proper mixing of objects from both senses of a verb in training and testing sets, we set the cross-validation process as follows: We first split both sets of objects in 4 parts. For each fold then, our training set is comprised by $\frac{3}{4}$ of set #1 plus $\frac{3}{4}$ of set #2, while the test set consists of the remaining $\frac{1}{4}$ of set #1 plus $\frac{1}{4}$ of set #2. The data points of the training set are presented in the

	Accuracy		MRR		Avg Sim	
	Amb.	Dis.	Amb.	Dis.	Amb.	Dis.
break	0.19	0.28	0.41	0.50	0.41	0.43
catch	0.35	0.37	0.58	0.61	0.51	0.57
play	0.20	0.28	0.41	0.49	0.60	0.68
admit	0.33	0.43	0.57	0.64	0.41	0.46
draw	0.24	0.29	0.45	0.51	0.40	0.44

Table 2: Results for the supervised task. ‘Amb.’ refers to models without the explicit disambiguation step, and ‘Dis.’ to models with that step.

learning algorithm in random order.

We measure approximation in three different metrics. The first one, accuracy, is the strictest, and evaluates in how many cases the composite vector of a verb phrase is the closest one (the first one in the result list) to the corresponding holistic vector. A more relaxed and perhaps more representative method is to calculate the mean reciprocal rank (MRR), which is given by:

$$\text{MRR} = \frac{1}{m} \sum_{i=1}^m \frac{1}{\text{rank}_i} \quad (6)$$

where m is the number of objects and rank_i refers to the rank of the correct composite vector for the i th object.

Finally, a third way to evaluate the efficiency of each model is to simply calculate the average cosine similarity between every holistic vector and its corresponding composite vector. The results are presented in Table 2, reflecting a clear superiority ($p < 0.001$ for average cosine similarity) of the prior disambiguation method for every verb and every metric.

7 Unsupervised disambiguation

In Section 6 we used a controlled procedure to collect genuinely ambiguous verbs and we trained our models from manually annotated data. In this section we briefly outline how the process of creating tensors for distinct senses of a verb can be automated, and we test this idea on a generic verb phrase similarity task.

First, we use unsupervised learning in order to detect the latent senses of each verb in the corpus, following a procedure first described by Schütze (1998). For every occurrence of the verb, we create a vector representing the surrounding context by averaging the vectors of every other word in the same sentence. Then, we apply hierarchical agglomerative clustering (HAC) in order to cluster these context vectors, hoping that different groups of contexts will correspond to the different senses under which the word has been used in the corpus. The clustering algorithm uses Ward’s method as

inter-cluster measure, and Pearson correlation for measuring the distance of vectors within a cluster. Since HAC returns a dendrogram embedding all possible groupings, we measure the quality of each partitioning by using the variance ratio criterion (Caliński and Harabasz, 1974) and we select the partitioning that achieves the best score (so the number of senses varies from verb to verb).

The next step is to classify every noun that has been used as an object with that verb to the most probable verb sense, and then use these sets of nouns as before for training tensors for the various verb senses. Being equipped with a number of sense clusters created as above for every verb, the classification of each object to a relevant sense is based on the cosine distance of the object vector from the centroids of the clusters.¹ Every sense with less than 3 training exemplars is merged to the dominant sense of the verb. The union of all object sets is used for training a single unambiguous tensor for the verb. As usual, data points are presented to learning algorithm in random order. No objects in our test set are used for training.

We test this system on a verb phase similarity task introduced in (Mitchell and Lapata, 2010). The goal is to assess the similarity between pairs of short verb phrases (verb-object constructs) and evaluate the results against human annotations. The dataset consists of 72 verb phrases, paired in three different ways to form groups of various degrees of phrase similarity—a total of 108 verb phrase pairs.

The experiment has the following form: For every pair of verb phrases, we construct composite vectors and then we evaluate their cosine similarity. For the ambiguous regression model, the composition is done by matrix-multiplying the ambiguous verb matrix (learned by the union of all object sets) with the vector of the noun. For the disambiguated version, we first detect the most probable sense of the verb given the noun, again by comparing the vector of the noun with the centroids of the verb clusters; then, we matrix-multiply the corresponding unambiguous tensor created exclusively from objects that have been classified as closer to this specific sense of the verb with the noun. We also test a number of baselines: the ‘verbs-only’ model is a non-compositional baseline where only the two verbs are compared; ‘additive’ and ‘multiplicative’ compose the word vectors of each phrase by applying simple element-wise operations.

¹In general, our approach is quite close to the multi-prototype models of Reisinger and Mooney (2010).

Model	Spearman’s ρ
Verbs-only	0.331
Additive	0.379
Multiplicative	0.301
Linear regression (ambiguous)	0.349
Linear regression (disamb.)	0.399
Holistic verb phrase vectors	0.403
Human agreement	0.550

Table 3: Results for the phrase similarity task. The difference between the ambiguous and the disambiguated version is s.s. with $p < 0.001$.

The results are presented in Table 3, where again the version with the prior disambiguation step shows performance superior to that of the ambiguous version. There are two interesting observations that can be made on the basis of Table 3. First of all, the regression model is based on the assumption that the holistic vectors of the exemplar verb phrases follow an ideal distributional behaviour that the model aims to approximate as close as possible. The results of Table 3 confirm this: using just the holistic vectors of the corresponding verb phrases (no composition is involved here) returns the best correlation with human annotations (0.403), providing a proof that the holistic vectors of the verb phrases are indeed reliable representations of each verb phrase’s meaning. Next, observe that the prior disambiguation model approximates this behaviour very closely (0.399) on unseen data, with a difference *not* statistically significant. This is very important, since a regression model can only perform as well as its training dataset allows it; and in our case this is achieved to a very satisfactory level.

8 Conclusion and future work

This paper adds to existing evidence from previous research that the introduction of an explicit disambiguation step before the composition improves the quality of the produced composed representations. The use of a robust regression model rejects the hypothesis that the proposed methodology is helpful only for relatively “weak” compositional approaches. As for future work, an interesting direction would be to see how a prior disambiguation step can affect deep learning compositional settings similar to (Socher et al., 2012) and (Kalchbrenner and Blunsom, 2013b).

Acknowledgements

We would like to thank the three anonymous reviewers for their fruitful comments. Support by EPSRC grant EP/F042728/1 is gratefully acknowledged by D. Kartsaklis and M. Sadrzadeh.

References

- M. Baroni and R. Zamparelli. 2010. Nouns are Vectors, Adjectives are Matrices. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- T. Caliński and J. Harabasz. 1974. A Dendrite Method for Cluster Analysis. *Communications in Statistics-Theory and Methods*, 3(1):1–27.
- B. Coecke, M. Sadrzadeh, and S. Clark. 2010. Mathematical Foundations for Distributed Compositional Model of Meaning. Lambek Festschrift. *Linguistic Analysis*, 36:345–384.
- Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukWaC, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google*, pages 47–54.
- Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-step regression learning for compositional distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)*.
- N. Kalchbrenner and P. Blunsom. 2013a. Recurrent convolutional neural networks for discourse compositionality. In *Proceedings of the 2013 Workshop on Continuous Vector Space Models and their Compositionality*, Sofia, Bulgaria, August.
- Nal Kalchbrenner and Phil Blunsom. 2013b. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Seattle, USA, October. Association for Computational Linguistics.
- Dimitri Kartsaklis and Mehrnoosh Sadrzadeh. 2013. Prior disambiguation of word tensors for constructing sentence vectors. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Seattle, USA, October.
- D. Kartsaklis, M. Sadrzadeh, and S. Pulman. 2013. Separating Disambiguation from Composition in Distributional Semantics. In *Proceedings of 17th Conference on Computational Natural Language Learning (CoNLL-2013)*, Sofia, Bulgaria, August.
- Dimitri Kartsaklis. 2014. Compositional operators in distributional semantics. *Springer Science Reviews*, April. DOI: 10.1007/s40362-014-0017-z.
- J. Mitchell and M. Lapata. 2008. Vector-based Models of Semantic Composition. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 236–244.
- Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1439.
- Siva Reddy, Ioannis Klapaftis, Diana McCarthy, and Suresh Manandhar. 2011. Dynamic and static prototype vectors for semantic composition. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 705–713.
- Joseph Reisinger and Raymond J Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117. Association for Computational Linguistics.
- H. Rubenstein and J.B. Goodenough. 1965. Contextual Correlates of Synonymy. *Communications of the ACM*, 8(10):627–633.
- R. Sansome, D. Reid, and A. Spooner. 2000. *The Oxford Junior Dictionary*. Oxford University Press.
- H. Schütze. 1998. Automatic Word Sense Discrimination. *Computational Linguistics*, 24:97–123.
- R. Socher, E.H. Huang, J. Pennington, A.Y. Ng, and C.D. Manning. 2011. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. *Advances in Neural Information Processing Systems*, 24.
- R. Socher, B. Huval, C. Manning, and Ng. A. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Conference on Empirical Methods in Natural Language Processing 2012*.

A Novel Content Enriching Model for Microblog Using News Corpus

Yunlun Yang¹, Zhihong Deng^{2*}, Hongliang Yu³

Key Laboratory of Machine Perception (Ministry of Education),
School of Electronics Engineering and Computer Science,
Peking University, Beijing 100871, China

¹incomparable-lun@pku.edu.cn

²zhdeng@cis.pku.edu.cn

³yuhongliang324@gmail.com

Abstract

In this paper, we propose a novel model for enriching the content of microblogs by exploiting external knowledge, thus improving the data sparseness problem in short text classification. We assume that microblogs share the same topics with external knowledge. We first build an optimization model to infer the topics of microblogs by employing the topic-word distribution of the external knowledge. Then the content of microblogs is further enriched by relevant words from external knowledge. Experiments on microblog classification show that our approach is effective and outperforms traditional text classification methods.

1 Introduction

During the past decade, the short text representation has been intensively studied. Previous researches (Phan et al., 2008; Guo and Diab, 2012) show that while traditional methods are not so powerful due to the data sparseness problem, some semantic analysis based approaches are proposed and proved effective, and various topic models are among the most frequently used techniques in this area. Meanwhile, external knowledge has been found helpful (Hu et al., 2009) in tackling the data scarcity problem by enriching short texts with informative context. Well-organized knowledge bases such as Wikipedia and WordNet are common tools used in relevant methods.

Nowadays, most of the work on short text focuses on microblog. As a new form of short text, microblog has some unique features like informal spelling and emerging words, and many microblogs are strongly related to up-to-date topics as well. Every day, a great quantity of microblogs

more than we can read is pushed to us, and finding what we are interested in becomes rather difficult, so the ability of choosing what kind of microblogs to read is urgently demanded by common user. Such ability can be implemented by effective short text classification.

Treating microblogs as standard texts and directly classifying them cannot achieve the goal of effective classification because of sparseness problem. On the other hand, news on the Internet is of information abundance and many microblogs are news-related. They share up-to-date topics and sometimes quote each other. Thus, external knowledge, such as news, provides rich supplementary information for analysing and mining microblogs.

Motivated by the idea of using topic model and external knowledge mentioned above, we present an LDA-based enriching method using the news corpus, and apply it to the task of microblog classification. The basic assumption in our model is that news articles and microblogs tend to share the same topics. We first infer the topic distribution of each microblog based on the topic-word distribution of news corpus obtained by the LDA estimation. With the above two distributions, we then add a number of words from news as additional information to microblogs by evaluating the relatedness of between each word and microblog, since words not appearing in the microblog may still be highly relevant.

To sum up, our contributions are:

- (1) We formulate the topic inference problem for short texts as a convex optimization problem.
- (2) We enrich the content of microblogs by inferring the association between microblogs and external words in a probabilistic perspective.
- (3) We evaluate our method on the real datasets and experiment results outperform the baseline methods.

*Corresponding author

2 Related Work

Based on the idea of exploiting external knowledge, many methods are proposed to improve the representation of short texts for classification and clustering. Among them, some directly utilize the structure information of organized knowledge base or search engine. Banerjee et al. (2007) use the title and the description of news article as two separate query strings to select related concepts as additional feature. Hu et al. (2009) present a framework to improve the performance of short text clustering by mining informative context with the integration of Wikipedia and WordNet.

However, to better leverage external resource, some other methods introduce topic models. Phan et al. (2008) present a framework including an approach for short text topic inference and adds abstract words as extra feature. Guo and Diab (2012) modify classic topic models and propose a matrix-factorization based model for sentence similarity calculation tasks.

Those methods without topic model usually rely greatly on the performance of search system or the completeness of knowledge base, and lack in-depth analysis for external resources. Compared with our method, the topic model based methods mentioned above remain in finding latent space representation of short text and ignore that relevant words from external knowledge are informative as well.

3 Our Model

We formulate the problem as follows. Let $EK = \{d_1^e, \dots, d_{M^e}^e\}$ denote external knowledge consisting of M^e documents. $V^e = \{w_1^e, \dots, w_{N^e}^e\}$ represents its vocabulary. Let $MB = \{d_1^m, \dots, d_{M^m}^m\}$ denote microblog set and its vocabulary is $V^m = \{w_1^m, \dots, w_{N^m}^m\}$. Our task is to enrich each microblog with additional information so as to improve microblog's representation.

The model we proposed mainly consists of three steps:

- (a) Topic inference for external knowledge by running LDA estimation.
- (b) Topic inference for microblogs by employing the word distributions of topics obtained from step (a).

- (c) Select relevant words from external knowledge to enrich the content of microblogs.

3.1 Topic Inference for External Knowledge

We do topic analysis for EK using LDA estimation (Blei et al., 2003) in this section and we choose LDA as the topic analysis model because of its broadly proved effectivity and ease of understanding.

In LDA, each document has a distribution over all topics $P(z_k|d_j)$, and each topic has a distribution over all words $P(w_i|z_k)$, where z_k , d_j and w_i represent the topic, document and word respectively. The optimization problem is formulated as maximizing the log likelihood on the corpus:

$$\max \sum_i \sum_j X_{ij} \log \sum_k P(z_k|d_j)P(w_i|z_k) \quad (1)$$

In this formulation, X_{ij} represents the term frequency of word w_i in document d_j . $P(z_k|d_j)$ and $P(w_i|z_k)$ are parameters to be inferred, corresponding to the topic distribution of each document and the word distribution of each topic respectively. Estimating parameters for LDA by directly and exactly maximizing the likelihood of the corpus in (1) is intractable, so we use Gibbs Sampling for estimation.

After performing LDA model (K topics) estimation on EK , we obtain the topic distributions of document d_j^e ($j = 1, \dots, M^e$), denoted as $P(z_k^e|d_j^e)$ ($k = 1, \dots, K$), and the word distribution of topic z_k^e ($k = 1, \dots, K$), denoted as $P(w_i^e|z_k^e)$ ($i = 1, \dots, N^e$). Step (b) greatly relies on the word distributions of topics we have obtained here.

3.2 Topic Inference for Microblog

In this section, we infer the topic distribution of each microblog. Because of the assumption that microblogs share the same topics with external corpus, the "topic distribution" here refers to a distribution over all topics on EK .

Differing from step (a), the method used for topic inference for microblogs is not directly running LDA estimation on microblog collection but following the topics from external knowledge to ensure topic consistence. We employ the word distributions of topics obtained from step (a), i.e. $P(w_i^e|z_k^e)$, and formulate the optimization problem in a similar form to Formula (1) as follows:

$$\max_{P(z_k^e|d_j^m)} \sum_i \sum_j \underline{X}_{ij} \log \sum_k P(z_k^e|d_j^m) P(w_i^e|z_k^e), \quad (2)$$

where \underline{X}_{ij} represents the term frequency of word w_i^e in microblog d_j^m , and $P(z_k^e|d_j^m)$ denote the distribution of microblog d_j^m over all topics on EK . Obviously most \underline{X}_{ij} are zero and we ignore those words that do not appear in V^e .

Compared with the original LDA optimization problem (1), the topic inference problem for microblog (2) follows the idea of document generation process, but replaces topics to be estimated with known topics from other corpus. As a result, parameters to be inferred are only the topic distribution of every microblog.

It is noteworthy that since the word distribution of every topic $P(w_i^e|z_k^e)$ is known, Formula (2) can be further solved by separating it into M^m sub-problems:

$$\max_{P(z_k^e|d_j^m)} \sum_i \underline{X}_{ij} \log \sum_k P(z_k^e|d_j^m) P(w_i^e|z_k^e) \quad \text{for } j = 1, \dots, M^m \quad (3)$$

These M^m subproblems correspond to the M^m microblogs and can be easily proved convexity. After solving them, we obtain the topic distributions of microblog d_j^m ($j = 1, \dots, M^m$), denoted as $P(z_k^e|d_j^m)$ ($k = 1, \dots, K$).

3.3 Select Relevant Words for Microblog

To enrich the content of every microblog, we select relevant words from external knowledge in this section.

Based on the results of step (a)&(b), we calculate the word distributions of microblogs as follows:

$$P(w_i^e|d_j^m) = \sum_k P(z_k^e|d_j^m) P(w_i^e|z_k^e), \quad (4)$$

where $P(w_i^e|d_j^m)$ represents the probability that word w_i^e will appear in microblog d_j^m . In other words, though some words may not actually appear in a microblog, there is still a probability that it is highly relevant to the microblog. Intuitively, this probability indicates the strength of association between a word and a microblog. The word

distribution of every microblog is based on topic analysis and its accuracy relies heavily on the accuracy of topic inference in step (b). In fact, the more words a microblog includes, the more accurate its topic inference will be, and this can be regarded as an explanation of the low efficiency of data sparseness problem.

For microblog d_j^m , we sort all words by $P(w_i^e|d_j^m)$ in descending order. Having known the top L relevant words according to the result of sorting, we redefine the ‘‘term frequency’’ of every word after adding these L words to microblog d_j^m as additional content. Supposing these L words are $w_{j1}^e, w_{j2}^e, \dots, w_{jL}^e$, the revised term frequency of word $w \in \{w_{j1}^e, \dots, w_{jL}^e\}$ is defined as follows:

$$RTF(w, d_j^m) = \frac{P(w|d_j^m)}{\sum_{p=1}^L P(w_{jp}^e|d_j^m)} * L, \quad (5)$$

where $RTF(\cdot)$ is the revised term frequency.

As the Equation (5) shows, the revised term frequency of every word is proportional to probability $P(w_i|d_j^m)$ rather than a constant.

So far, we can add these L **words and their revised term frequency** as additional information to microblog d_j^m . The revised term frequency plays the same role as TF in common text representation vector, so we calculate the TFIDF of the added words as:

$$TFIDF(w, d_j^m) = RTF(w, d_j^m) \cdot IDF(w) \quad (6)$$

Note that $IDF(w)$ is changed as arrival of new words for each microblog. The TFIDF vector of a microblog with additional words is called **enhanced vector**.

4 Experiment

4.1 Experimental Setup

To evaluate our method, we build our own datasets. We crawl 95028 Chinese news reports from Sina News website, segment them, and remove stop words and rare words. After preprocessing, these news documents are used as external knowledge. As for microblog, we crawl a number of microblogs from Sina Weibo, and ask unbiased assessors to manually classify them into 9 categories following the column setting of Sina News.

Sina News: <http://news.sina.com.cn/>
Sina Weibo: <http://www.weibo.com/>

After the manual classification, we remove short microblogs (less than 10 words), usernames, links and some special characters, then we segment them and remove rare words as well. Finally, we get 1671 classified microblogs as our microblog dataset. The size of each category is shown in Table 1.

Category	#Microblog
Finance	229
Stock	80
Entertainment	162
Military Affairs	179
Technologies	204
Digital Products	194
Sports	195
Society	214
Daily Life	214

Table 1: Microblog number of every category

There are some important details of our implementation. In step (a) of Section 3.1 we estimate LDA model using GibbsLDA++, a C/C++ implementation of LDA using Gibbs Sampling. In step (b) of Section 3.2, OPTI toolbox on Matlab is used to help solve the convex problems. In the classification tasks shown below, we use LibSVM as classifier and perform ten-fold cross validation to evaluate the classification accuracy.

4.2 Classification Results

Representation	Average Accuracy
TFIDF vector	0.7552
Boolean vector	0.7203
Enhanced vector	0.8453

Table 2: Classification accuracy with different representations

In this section, we report the average precision of each method as shown in Table 2. The *enhanced vector* is the representation generated by our method. Two baselines are *TFIDF vector* (Jones, 1972) and *boolean vector* (word occurrence) of the original microblog. In the table, our method increases the classification accuracy

GibbsLDA++: <http://gibbslda.sourceforge.net>
OPTI Toolbox: <http://www.i2c2.aut.ac.nz/Wiki/OPTI/>
SVM.NET: <http://www.matthewajohnson.org/software/svm.html>

from 75.52% to 84.53% when considering additional information, which means our method indeed improves the representation of microblogs.

4.3 Parameter Tuning

4.3.1 Effect of Added Words

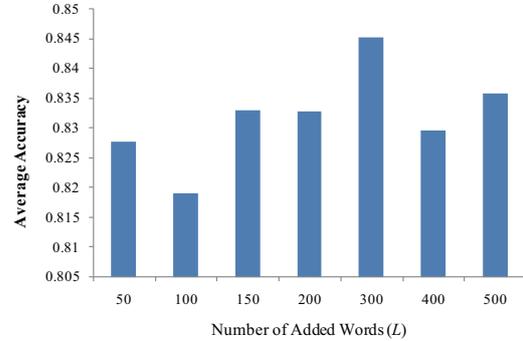


Figure 1: Classification accuracy changes according to topics and added words

The experiment corresponding to Figure 1 is to discover how the classification accuracy changes when we fix the number of topics ($K = 100$) and change the number of added words (L) in our method. Result shows that more added words do not mean higher accuracy. By studying some cases, we find out that if we add too many words, the proportion of “noisy words” will increase. We reach the best result when number of added words is 300.

4.3.2 Effect of Topic Number

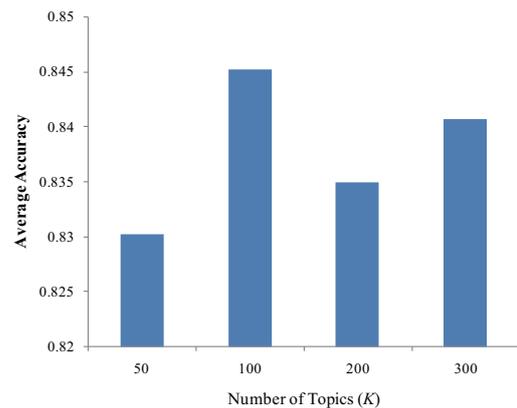


Figure 2: Classification accuracy changing according to the number of topics

The experiment corresponding to Figure 2 is to discover how the classification accuracy changes when we fix the number of added words ($L =$

Microblog (Translated)	Top Relevant Words (Translated)
Kim Jong Un held an emergency meeting this morning, and commanded the missile units to prepare for attacking U.S. military bases at any time.	South Korea, America, North Korea, work, safety, claim, military, exercise, united, report
Shenzhou Nine will carry three astronauts, including the first Chinese female astronaut, and launch in a proper time during the middle of June.	day, satellite, launch, research, technology, system, mission, aerospace, success, Chang'e Two

Table 3: Case study (Translated from Chinese)

300) and change the number of topics (K) in our method. As we can see, the accuracy does not grow monotonously as the number of topics increases. Blindly enlarging the topic number will not improve the accuracy. The best result is reached when topic number is 100, and similar experiments adding different number of words show the same condition of reaching the best result.

4.3.3 Effect of Revised Term Frequency

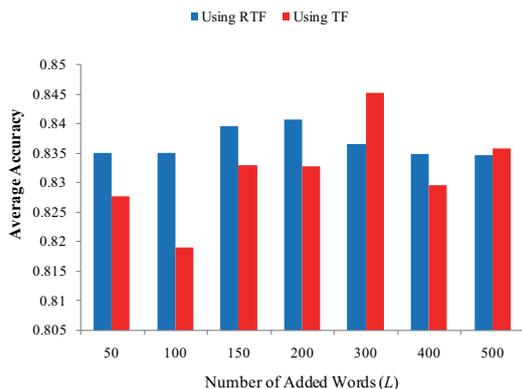


Figure 3: Classification accuracy changing according to the redefinition of term frequency

The experiment corresponding to Figure 3 is to discover whether our redefining “term frequency” as revised term frequency in step (c) of Section 3.3 will affect the classification accuracy and how. The results should be analysed in two aspects. On one hand, without redefinition, the accuracy remains in a stable high level and tends to decrease as we add more words. One reason for the decreasing is that “noisy words” have a increasing negative impact on the accuracy as the proportion of “noisy words” grows with the number of added words. On the other hand, the best result is reached when we use the revise term frequency. This suggests that our redefinition for term frequency shows better improvement for microblog

representation under certain conditions, but is not optimal under all situations.

4.4 Case Study

In Table 3, we select several cases consisting of microblogs and their top relevant words .

In the first case, we successfully find the country name according to its leader’s name and limited information in the sentence. Other related countries and events are also selected by our model as they often appear together in news. In the other case, relevant words are among the most frequently used words in news and have close semantic relations with the microblogs in certain aspects.

As we can see, based on topic analysis, our model shows strong ability of mining relevant words. Other cases show that the model can be further improved by removing the noisy and meaningless ones among added words.

5 Conclusion and Future Work

We propose an effective content enriching method for microblog, to enhance classification accuracy. News corpus is exploited as external knowledge. As for techniques, our method uses LDA as its topic analysis model and formulates topic inference for new data as convex optimization problems. Compared with traditional representation, enriched microblog shows great improvement in classification tasks.

As we do not control the quality of added words, our future work starts from building a filter to select better additional information. And to make the most of external knowledge, better ways to build topic space should be considered.

Acknowledgments

This work is supported by National Natural Science Foundation of China (Grant No. 61170091).

References

- Banerjee, S., Ramanathan, K., and Gupta, A. 2007, July. Clustering short texts using wikipedia. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 787-788). ACM.
- Blei, D. M., Ng, A. Y., and Jordan, M. I. 2003. Latent Dirichlet Allocation. In *Journal of machine Learning research*, 3, 993-1022.
- Bollegala, D., Matsuo, Y., and Ishizuka, M. 2007. Measuring semantic similarity between words using web search engines. *www*, 7, 757-766.
- Boyd, S. P., and Vandenberghe, L. 2004. *Convex optimization*. Cambridge university press.
- Gabrilovich, E., and Markovitch, S. 2007, January. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *IJCAI* (Vol. 7, pp. 1606-1611).
- Guo, W., and Diab, M. 2012, July. Modeling sentences in the latent space. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1* (pp. 864-872).
- Guo, W., and Diab, M. 2012, July. Learning the latent semantics of a concept from its definition. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2* (pp. 140-144).
- Hu, X., Sun, N., Zhang, C., and Chua, T. S. 2009, November. Exploiting internal and external semantics for the clustering of short texts using world knowledge. In *Proceedings of the 18th ACM conference on Information and knowledge management* (pp. 919-928). ACM.
- Jones, K. S. 1972. A statistical interpretation of term specificity and its application in retrieval. In *Journal of documentation*, 28(1), 11-21
- Phan, X. H., Nguyen, L. M., and Horiguchi, S. 2008, April. Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections. In *Proceedings of the 17th international conference on World Wide Web* (pp. 91-100). ACM.
- Sahami, M., and Heilman, T. D. 2006, May. A web-based kernel function for measuring the similarity of short text snippets. In *Proceedings of the 15th international conference on World Wide Web* (pp. 377-386). ACM.
- Zubiaga, A., and Ji, H. 2013, May. Harnessing web page directories for large-scale classification of tweets. In *Proceedings of the 22nd international conference on World Wide Web companion* (pp. 225-226). International World Wide Web Conferences Steering Committee.

Learning Bilingual Word Representations by Marginalizing Alignments

Tomáš Kočiský

Karl Moritz Hermann

Phil Blunsom

Department of Computer Science
University of Oxford
Oxford, OX1 3QD, UK

{tomas.kocisky, karl.moritz.hermann, phil.blunsom}@cs.ox.ac.uk

Abstract

We present a probabilistic model that simultaneously learns alignments and distributed representations for bilingual data. By marginalizing over word alignments the model captures a larger semantic context than prior work relying on hard alignments. The advantage of this approach is demonstrated in a cross-lingual classification task, where we outperform the prior published state of the art.

1 Introduction

Distributed representations have become an increasingly important tool in machine learning. Such representations—typically continuous vectors learned in an unsupervised setting—can frequently be used in place of hand-crafted, and thus expensive, features. By providing a richer representation than what can be encoded in discrete settings, distributed representations have been successfully used in many areas. This includes AI and reinforcement learning (Mnih et al., 2013), image retrieval (Kiros et al., 2013), language modelling (Bengio et al., 2003), sentiment analysis (Socher et al., 2011; Hermann and Blunsom, 2013), frame-semantic parsing (Hermann et al., 2014), and document classification (Klementiev et al., 2012).

In Natural Language Processing (NLP), the use of distributed representations is motivated by the idea that they could capture semantics and/or syntax, as well as encoding a continuous notion of similarity, thereby enabling information sharing between similar words and other units. The success of distributed approaches to a number of tasks, such as listed above, supports this notion and its implied benefits (see also Turian et al. (2010) and Collobert and Weston (2008)).

While most work employing distributed representations has focused on monolingual tasks, multilingual representations would also be useful for

several NLP-related tasks. Such problems include document classification, machine translation, and cross-lingual information retrieval, where multilingual data is frequently the norm. Furthermore, learning multilingual representations can also be useful for cross-lingual information transfer, that is exploiting resource-fortunate languages to generate supervised data in resource-poor ones.

We propose a probabilistic model that simultaneously learns word alignments and bilingual distributed word representations. As opposed to previous work in this field, which has relied on hard alignments or bilingual lexica (Klementiev et al., 2012; Mikolov et al., 2013), we marginalize out the alignments, thus capturing more bilingual semantic context. Further, this results in our distributed word alignment (DWA) model being the first probabilistic account of bilingual word representations. This is desirable as it allows better reasoning about the derived representations and furthermore, makes the model suitable for inclusion in higher-level tasks such as machine translation.

The contributions of this paper are as follows. We present a new probabilistic similarity measure which is based on an alignment model and prior language modeling work which learns and relates word representations across languages. Subsequently, we apply these embeddings to a standard document classification task and show that they outperform the current published state of the art (Hermann and Blunsom, 2014b). As a by-product we develop a distributed version of FASTALIGN (Dyer et al., 2013), which performs on par with the original model, thereby demonstrating the efficacy of the learned bilingual representations.

2 Background

The IBM alignment models, introduced by Brown et al. (1993), form the basis of most statistical machine translation systems. In this paper we base our alignment model on FASTALIGN (FA), a vari-

ation of IBM model 2 introduced by Dyer et al. (2013). This model is both fast and produces alignments on par with the state of the art. Further, to induce the distributed representations we incorporate ideas from the log-bilinear language model presented by Mnih and Hinton (2007).

2.1 IBM Model 2

Given a parallel corpus with aligned sentences, an alignment model can be used to discover matching words and phrases across languages. Such models are an integral part of most machine translation pipelines. An alignment model learns $p(\mathbf{f}, \mathbf{a}|\mathbf{e})$ (or $p(\mathbf{e}, \mathbf{a}'|\mathbf{f})$) for the source and target sentences \mathbf{e} and \mathbf{f} (sequences of words). \mathbf{a} represents the word alignment across these two sentences from source to target. IBM model 2 (Brown et al., 1993) learns alignment and translation probabilities in a generative style as follows:

$$p(\mathbf{f}, \mathbf{a}|\mathbf{e}) = p(J|I) \prod_{j=1}^J p(a_j|j, I, J) p(f_j|e_{a_j}),$$

where $p(J|I)$ captures the two sentence lengths; $p(a_j|j, I, J)$ the alignment and $p(f_j|e_{a_j})$ the translation probability. Sentence likelihood is given by marginalizing out the alignments, which results in the following equation:

$$p(\mathbf{f}|\mathbf{e}) = p(J|I) \prod_{j=1}^J \sum_{i=0}^I p(i|j, I, J) p(f_j|e_i).$$

We use FASTALIGN (FA) (Dyer et al., 2013), a log-linear reparametrization of IBM model 2. This model uses an alignment distribution defined by a single parameter that measures how close the alignment is to the diagonal. This replaces the original multinomial alignment distribution which often suffered from sparse counts. This improved model was shown to run an order of magnitude faster than IBM model 4 and yet still outperformed it in terms of the BLEU score and, on Chinese-English data, in alignment error rate (AER).

2.2 Log-Bilinear Language Model

Language models assign a probability measure to sequences of words. We use the log-bilinear language model proposed by Mnih and Hinton (2007). It is an n-gram based model defined in terms of an energy function $E(w_n; w_{1:n-1})$. The probability for predicting the next word w_n given its preceding context of $n - 1$ words is expressed

using the energy function

$$E(w_n; w_{1:n-1}) = - \left(\sum_{i=1}^{n-1} r_{w_i}^T C_i \right) r_{w_n} - b_r^T r_{w_n} - b_{w_n}$$

as $p(w_n|w_{1:n-1}) = \frac{1}{Z_c} \exp(-E(w_n; w_{1:n-1}))$ where $Z_c = \sum_{w_n} \exp(-E(w_n; w_{1:n-1}))$ is the normalizer, $r_{w_i} \in \mathbb{R}^d$ are word representations, $C_i \in \mathbb{R}^{d \times d}$ are context transformation matrices, and $b_r \in \mathbb{R}^d$, $b_{w_n} \in \mathbb{R}$ are representation and word biases respectively. Here, the sum of the transformed context-word vectors endeavors to be close to the word we want to predict, since the likelihood in the model is maximized when the energy of the observed data is minimized.

This model can be considered a variant of a log-linear language model in which, instead of defining binary n-gram features, the model learns the features of the input and output words, and a transformation between them. This provides a vastly more compact parameterization of a language model as n-gram features are not stored.

2.3 Multilingual Representation Learning

There is some recent prior work on multilingual distributed representation learning. Similar to the model presented here, Klementiev et al. (2012) and Zou et al. (2013) learn bilingual embeddings using word alignments. These two models are non-probabilistic and conditioned on the output of a separate alignment model, unlike our model, which defines a probability distribution over translations and marginalizes over all alignments. These models are also highly related to prior work on bilingual lexicon induction (Haghighi et al., 2008). Other recent approaches include Sarath Chandar et al. (2013), Lauly et al. (2013) and Hermann and Blunsom (2014a, 2014b). These models avoid word alignment by transferring information across languages using a composed sentence-level representation.

While all of these approaches are related to the model proposed in this paper, it is important to note that our approach is novel by providing a probabilistic account of these word embeddings. Further, we learn word alignments and simultaneously use these alignments to guide the representation learning, which could be advantageous particularly for rare tokens, where a sentence based approach might fail to transfer information.

Related work also includes Mikolov et al. (2013), who learn a transformation matrix to

reconcile monolingual embedding spaces, in an l_2 norm sense, using dictionary entries instead of alignments, as well as Schwenk et al. (2007) and Schwenk (2012), who also use distributed representations for estimating translation probabilities. Faruqui and Dyer (2014) use a technique based on CCA and alignments to project monolingual word representations to a common vector space.

3 Model

Here we describe our distributed word alignment (DWA) model. The DWA model can be viewed as a distributed extension of the FA model in that it uses a similarity measure over distributed word representations instead of the standard multinomial translation probability employed by FA. We do this using a modified version of the log-bilinear language model in place of the translation probabilities $p(f_j|e_i)$ at the heart of the FA model. This allows us to learn word representations for both languages, a translation matrix relating these vector spaces, as well as alignments at the same time.

Our modifications to the log-bilinear model are as follows. Where the original log-bilinear language model uses context words to predict the next word—this is simply the distributed extension of an n-gram language model—we use a word from the source language in a parallel sentence to predict a target word. An additional aspect of our model, which demonstrates its flexibility, is that it is simple to include further context from the source sentence, such as words around the aligned word or syntactic and semantic annotations. In this paper we experiment with a transformed sum over k context words to each side of the aligned source word. We evaluate different context sizes and report the results in Section 5. We define the energy function for the translation probabilities to be

$$E(f, e_i) = - \left(\sum_{s=-k}^k r_{e_{i+s}}^T T_s \right) r_f - b_r^T r_f - b_f \quad (1)$$

where $r_{e_i}, r_f \in \mathbb{R}^d$ are vector representations for source and target words $e_{i+s} \in V_E, f \in V_F$ in their respective vocabularies, $T_s \in \mathbb{R}^{d \times d}$ is the transformation matrix for each surrounding context position, $b_r \in \mathbb{R}^d$ are the representation biases, and $b_f \in \mathbb{R}$ is a bias for each word $f \in V_F$.

The translation probability is given by $p(f|e_i) = \frac{1}{Z_{e_i}} \exp(-E(f, e_i))$, where $Z_{e_i} = \sum_f \exp(-E(f, e_i))$ is the normalizer.

In addition to these translation probabilities, we

have parameterized the translation probabilities for the null word using a softmax over an additional weight vector.

3.1 Class Factorization

We improve training performance using a class factorization strategy (Morin and Bengio, 2005) as follows. We augment the translation probability to be $p(f|e) = p(c_f|e)p(f|c_f, e)$ where c_f is a unique predetermined class of f ; the class probability is modeled using a similar log-bilinear model as above, but instead of predicting a word representation r_f we predict the class representation r_{c_f} (which is learned with the model) and we add respective new context matrices and biases. Note that the probability of the word f depends on *both* the class and the given context words: it is normalized only over words in the class c_f .

In our training we create classes based on word frequencies in the corpus as follows. Considering words in the order of their decreasing frequency, we add word types into a class until the total frequency of the word types in the currently considered class is less than $\frac{\text{total tokens}}{\sqrt{|V_F|}}$ and the class size is less than $\sqrt{|V_F|}$. We have found that the maximal class size affects the speed the most.

4 Learning

The original FA model optimizes the likelihood using the expectation maximization (EM) algorithm where, in the M-step, the parameter update is analytically solvable, except for the λ parameter (the diagonal tension), which is optimized using gradient descent (Dyer et al., 2013). We modified the implementations provided with CDEC (Dyer et al., 2010), retaining its default parameters.

In our model, DWA, we optimize the likelihood using the EM as well. However, while training we fix the counts of the E-step to those computed by FA, trained for the default 5 iterations, to aid the convergence rate, and optimize the M-step only. Let θ be the parameters for our model. Then the gradient for each sentence is given by

$$\frac{\partial}{\partial \theta} \log p(\mathbf{f}|\mathbf{e}) = \sum_{k=1}^J \sum_{l=0}^I \left[\frac{p(l|k, I, J) p(f_k|e_l)}{\sum_{i=0}^I p(i|k, I, J) p(f_k|e_i)} \cdot \frac{\partial}{\partial \theta} \log(p(l|k, I, J) p(f_k|e_l)) \right]$$

where the first part are the counts from the FA model and second part comes from our model.

We compute the gradient for the alignment probabilities in the same way as in the FA model, and the gradient for the translation probabilities using back-propagation (Rumelhart et al., 1986). For parameter update, we use ADAGRAD as the gradient descent algorithm (Duchi et al., 2011).

5 Experiments

We first evaluate the alignment error rate of our approach, which establishes the model’s ability to both learn alignments as well as word representations that explain these alignments. Next, we use a cross-lingual document classification task to verify that the representations are semantically useful. We also inspect the embedding space qualitatively to get some insight into the learned structure.

5.1 Alignment Evaluation

We compare the alignments learned here with those of the FASTALIGN model which produces very good alignments and translation BLEU scores. We use the same language pairs and datasets as in Dyer et al. (2013), that is the FBIS Chinese-English corpus, and the French-English section of the Europarl corpus (Koehn, 2005). We used the preprocessing tools from CDEC and further replaced all unique tokens with UNK. We trained our models with 100 dimensional representations for up to 40 iterations, and the FA model for 5 iterations as is the default.

Table 1 shows that our model learns alignments on par with those of the FA model. This is in line with expectation as our model was trained using the FA expectations. However, it confirms that the learned word representations are able to explain translation probabilities. Surprisingly, context seems to have little impact on the alignment error, suggesting that the model receives sufficient information from the aligned words themselves.

5.2 Document Classification

A standard task for evaluating cross-lingual word representations is document classification where training is performed in one and evaluation in another language. This tasks require semantically plausible embeddings (for classification) which are valid across two languages (for the semantic transfer). Hence this task requires more of the word embeddings than the previous task.

Languages	Model		
	FA	DWA $k = 0$	DWA $k = 3$
ZH EN	49.4	48.4	48.7
EN ZH	44.9	45.3	45.9
FR EN	17.1	17.2	17.0
EN FR	16.6	16.3	16.1

Table 1: Alignment error rate (AER) comparison, in both directions, between the FASTALIGN (FA) alignment model and our model (DWA) with k context words (see Equation 1). Lower numbers indicate better performance.

We mainly follow the setup of Klementiev et al. (2012) and use the German-English parallel corpus of the European Parliament proceedings to train the word representations. We perform the classification task on the Reuters RCV1/2 corpus. Unlike Klementiev et al. (2012), we do not use that corpus during the representation learning phase. We remove all words occurring less than five times in the data and learn 40 dimensional word embeddings in line with prior work.

To train a classifier on English data and test it on German documents we first project word representations from English into German: we select the most probable German word according to the learned translation probabilities, and then compute document representations by averaging the word representations in each document. We use these projected representations for training and subsequently test using the original German data and representations. We use an averaged perceptron classifier as in prior work, with the number of epochs (3) tuned on a subset of the training set.

Table 2 shows baselines from previous work and classification accuracies. Our model outperforms the model by Klementiev et al. (2012), and it also outperforms the most comparable models by Hermann and Blunsom (2014b) when training on German data and performs on par with it when training on English data.¹ It seems that our model learns more informative representations towards document classification, even without additional monolingual language models or context information. Again the impact of context is inconclusive.

¹From Hermann and Blunsom (2014a, 2014b) we only compare with models equivalent with respect to embedding dimensionality and training data. They still achieve the state of the art when using additional training data.

Model	en \rightarrow de	de \rightarrow en
Majority class	46.8	46.8
Glossed	65.1	68.6
MT	68.1	67.4
Klementiev et al.	77.6	71.1
BiCVM ADD	83.7	71.4
BiCVM BI	83.4	69.2
DWA ($k = 0$)	82.8	76.0
DWA ($k = 3$)	83.1	75.4

Table 2: Document classification accuracy when trained on 1,000 training examples of the RCV1/2 corpus (train \rightarrow test). Baselines are the majority class, glossed, and MT (Klementiev et al., 2012). Further, we are comparing to Klementiev et al. (2012), BiCVM ADD (Hermann and Blunsom, 2014a), and BiCVM BI (Hermann and Blunsom, 2014b). k is the context size, see Equation 1.

5.3 Representation Visualization

Following the document classification task we want to gain further insight into the types of features our embeddings learn. For this we visualize word representations using t-SNE projections (van der Maaten and Hinton, 2008). Figure 1 shows an extract from our projection of the 2,000 most frequent German words, together with an expected representation of a translated English word given translation probabilities. Here, it is interesting to see that the model is able to learn related representations for words *chair* and *ratspräsidentenschaft* (presidency) even though these words were not aligned by our model. Figure 2 shows an extract from the visualization of the 10,000 most frequent English words trained on another corpus. Here again, it is evident that the embeddings are semantically plausible with similar words being closely aligned.

6 Conclusion

We presented a new probabilistic model for learning bilingual word representations. This distributed word alignment model (DWA) learns both representations and alignments at the same time. We have shown that the DWA model is able to learn alignments on par with the FASTALIGN alignment model which produces very good alignments, thereby determining the efficacy of the learned representations which are used to calculate



Figure 1: A visualization of the expected representation of the translated English word *chair* among the nearest German words: words never aligned (green), and those seen aligned (blue) with it.

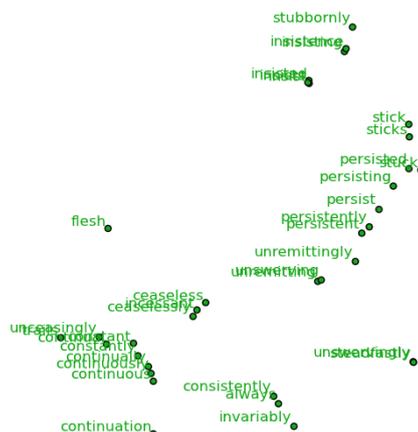


Figure 2: A cluster of English words from the 10,000 most frequent English words visualized using t-SNE. Word representations were optimized for $p(\text{zh}|\text{en})$ ($k = 0$).

word translation probabilities for the alignment task. Subsequently, we have demonstrated that our model can effectively be used to project documents from one language to another. The word representations our model learns as part of the alignment process are semantically plausible and useful. We highlighted this by applying these embeddings to a cross-lingual document classification task where we outperform prior work, achieve results on par with the current state of the art and provide new state-of-the-art results on one of the tasks. Having provided a probabilistic account of word representations across multiple languages, future work will focus on applying this model to machine translation and related tasks, for which previous approaches of learning such embeddings are less suited. Another avenue for further study is to combine this method with monolingual language models, particularly in the context of semantic transfer into resource-poor languages.

Acknowledgements

This work was supported by a Xerox Foundation Award and EPSRC grant number EP/K036580/1. We acknowledge the use of the Oxford ARC.

References

- Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, February.
- Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311, June.
- Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of ICML*.
- 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, July.
- Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of ACL System Demonstrations*.
- Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of NAACL-HLT*.
- Manaal Faruqui and Chris Dyer. 2014. Improving Vector Space Word Representations Using Multilingual Correlation. In *Proceedings of EACL*.
- Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL-HLT*.
- Karl Moritz Hermann and Phil Blunsom. 2013. The Role of Syntax in Vector Space Models of Compositional Semantics. In *Proceedings of ACL*.
- Karl Moritz Hermann and Phil Blunsom. 2014a. Multilingual Distributed Representations without Word Alignment. In *Proceedings of ICLR*.
- Karl Moritz Hermann and Phil Blunsom. 2014b. Multilingual Models for Compositional Distributional Semantics. In *Proceedings of ACL*.
- Karl Moritz Hermann, Dipanjan Das, Jason Weston, and Kuzman Ganchev. 2014. Semantic Frame Identification with Distributed Word Representations. In *Proceedings of ACL*.
- Ryan Kiros, Richard S Zemel, and Ruslan Salakhutdinov. 2013. Multimodal neural language models. In *NIPS Deep Learning Workshop*.
- Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. 2012. Inducing crosslingual distributed representations of words. In *Proceedings of COLING*.
- Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of the 10th Machine Translation Summit*.
- Stanislas Lauly, Alex Boulanger, and Hugo Larochelle. 2013. Learning multilingual word representations using a bag-of-words autoencoder. In *NIPS Deep Learning Workshop*.
- Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.
- Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of ICML*.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*.
- Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 246–252.
- D. E. Rumelhart, G. E. Hinton, and R. J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323:533–536, October.
- A P Sarath Chandar, M Khapra Mitesh, B Ravindran, Vikas Raykar, and Amrita Saha. 2013. Multilingual deep learning. In *Deep Learning Workshop at NIPS*.
- Holger Schwenk, Marta R. Costa-jussa, and Jose A. R. Fonollosa. 2007. Smooth bilingual n -gram translation. In *Proceedings of EMNLP-CoNLL*.
- Holger Schwenk. 2012. Continuous space translation models for phrase-based statistical machine translation. In *Proceedings of COLING: Posters*.
- Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of EMNLP*.
- Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of ACL*.
- L.J.P. van der Maaten and G.E. Hinton. 2008. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.
- Will Y. Zou, Richard Socher, Daniel Cer, and Christopher D. Manning. 2013. Bilingual Word Embeddings for Phrase-Based Machine Translation. In *Proceedings of EMNLP*.

Detecting Retries of Voice Search Queries

Rivka Levitan

Columbia University*

rlevitan@cs.columbia.edu

David Elson

Google Inc.

elson@google.com

Abstract

When a system fails to correctly recognize a voice search query, the user will frequently retry the query, either by repeating it exactly or rephrasing it in an attempt to adapt to the system's failure. It is desirable to be able to identify queries as retries both offline, as a valuable quality signal, and online, as contextual information that can aid recognition. We present a method that can identify retries offline with 81% accuracy using similarity measures between two subsequent queries as well as system and user signals of recognition accuracy. The retry rate predicted by this method correlates significantly with a gold standard measure of accuracy, suggesting that it may be useful as an offline predictor of accuracy.

1 Introduction

With ever more capable smartphones connecting users to cloud-based computing, voice has been a rapidly growing modality for searching for information online. Our voice search application connects a speech recognition service with a search engine, providing users with structured answers to questions, Web results, voice actions such as setting an alarm, and more. In the multimodal smartphone interface, users can press a button to activate the microphone, and then speak the query when prompted by a beep; after receiving results, the microphone button is available if they wish to follow up with a subsequent voice query.

Traditionally, the evaluation of speech recognition systems has been carried by preparing a test set of annotated utterances and comparing the accuracy of a system's transcripts of those utterances

*This work was done while the first author was an intern at Google Inc.

against the annotations. In particular, we seek to measure and minimize the word error rate (WER) of a system, with a WER of zero indicating perfect transcription. For voice search interfaces such as the present one, though, query-level metrics like WER only tell part of the story. When a user issues two queries in a row, she might be seeking the same information for a second time due to a system failure the first time. When this happens, from an evaluation standpoint it is helpful to break down why the first query was unsuccessful: it might be a speech recognition issue (in particular, a mistaken transcription), a search quality issue (where a correct transcript is interpreted incorrectly by the semantic understanding systems), a user interface issue, or another factor. As a second voice query may also be a new query or a follow-up query, as opposed to a retry of the first query, the detection of voice search retry pairs in the query stream is non-trivial.

Correctly identifying a retry situation in the query stream has two main benefits. The first involves offline evaluation and monitoring. We would like to know the rate at which users were forced to retry their voice queries, as a measure of quality. The second has a more immediate benefit for individual users: if we can detect in real time that a new voice search is really a retry of a previous voice search, we can take immediate corrective action, such as reranking transcription hypotheses to avoid making the same mistake twice, or presenting alternative searches in the user interface to indicate that the system acknowledges it is having difficulty.

In this paper, we describe a method for the *classification of subsequent voice searches* as either retry pairs of a certain type, or non-retry pairs. We identify four salient types of retry pairs, describe a test set and identify the features we extracted to build an automatic classifier. We then describe the models we used to build the classifier and their rel-

ative performance on the task, and leave the issue of real-time corrective action to future work.

2 Related Work

Previous work in voice-enabled information retrieval has investigated the problem of identifying voice retries, and some has taken the additional step of taking corrective action in instances where the user is thought to be retrying an earlier utterance. Zweig (2009) describes a system switching approach in which the second utterance is recognized by a separate model, one trained differently than the primary model. The “backup” system is found to be quite effective at recognizing those utterances missed by the primary system. Retry cases are identified with joint language modeling across multiple transcripts, with the intuition that retry pairs tend to be closely related or exact duplicates. They also propose a joint acoustic model in which portions of both utterances are averaged for feature extraction. Zweig et al. (2008) similarly create a joint decoding model under the assumption that a discrete set of entities (names of businesses with directory information) underlies both queries. While we follow this work in our usage of joint language modeling, our application encompasses open domain voice searches and voice actions (such as placing calls), so we cannot use simplifying domain assumptions.

Other approaches include Cevik, Weng and Lee (2008), who use dynamic time warping to define pattern boundaries using spectral features, and then consider the best matching patterns to be repeated. Williams (2008) measures the overlap between the two utterances’ n-best lists (alternate hypotheses) and upweights hypotheses that are common to both attempts; similarly, Orlandi, Culy and Franco (2003) remove hypotheses that are semantically equivalent to a previously rejected hypothesis. Unlike these approaches, we do not assume a strong notion of dialog state to maintain per-state models.

Another consequence of the open-domain nature of our service is that users are conditioned to interact with the system as they would with a search engine, e.g., if the results of a search do not satisfy their information need, they rephrase queries in order to refine their results. This can happen *even if the first transcript was correct* and the rephrased query can be easily confused for a retry of an utterance where the recognition failed.

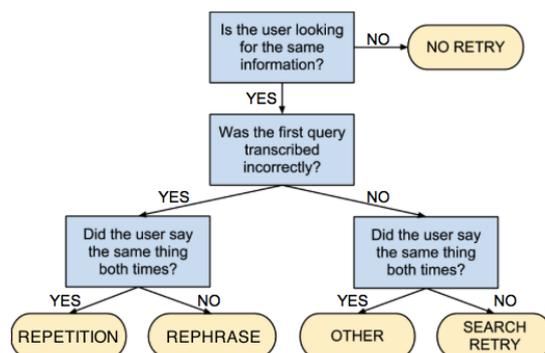


Figure 1: Retry annotation decision tree.

For purposes of latently monitoring the accuracy of the recognizer from usage logs, this is a significant complicating factor.

3 Data and Annotation

Our data consists of pairs of queries sampled from anonymized session logs. We consider a pair of voice searches (spoken queries) to be a potential retry pair if they are consecutive; we assume that a voice search cannot be a retry of another voice search if a typed search occurs between them. We also exclude pairs for which either member has no recognition result. For the purpose of our analysis, we further restricted our data to query pairs whose second member had been previously randomly selected for transcription. A set of 8,254 query pairs met these requirements and are considered potential retry pairs. 1,000 randomly selected pairs from this set were separated out and annotated by the authors, leaving a test set of 7,254 potential retry pairs. Among the annotated development set, 18 inaudible or unintelligible pairs were discarded, for a final development set of 982 pairs.

The problem as we have formulated it requires a labeling system that identifies repetitions and rephrases as retries, while excluding query pairs that are superficially similar but have different search intents. Our system includes five labels. Figure 1 shows the guidelines for annotation that define each category.

The first distinction is between query pairs with the same *search intent* (“Is the user looking for the same information?”) and those with different search intents. We define search intent as the response the user wants and expects from the system. If the second query’s search intent is different, it is by definition **no retry**.

The second distinction we make is between cases where the first query was recognized cor-

rectly and those where it was not. Although a query that was recognized correctly may be retried—for example, the user may want to be reminded of information she already received (**other**)—we are only interested in cases where the system is in error.

If the search intent is the same for both queries, and the system incorrectly recognized the first, we consider the second query a retry. We distinguish between cases where the user repeated the query exactly, **repetition**, and where the user rephrased the query in an attempt to adapt to the system’s failure, **rephrase**. This category includes many kinds of rephrasings, such as adding or dropping terms, or replacing them with synonyms. The rephrased query may be significantly different from the original, as in the following example:

Q1. *Navigate to chaparral ease.* (“Navigate to Chiapparelli’s.”)

Q2. *Chipper rally’s Little Italy Baltimore.* (“Chiapparelli’s Little Italy Baltimore.”)

The rephrased query dropped a term (“Navigate to”) and added another (“Little Italy Baltimore”).

This example illustrates another difficulty of the data: the unreliability of the automatic speech recognition (ASR) means that terms that are in fact identical (“Chiapparelli’s”) may be recognized very differently (“chaparral ease” or “chipper rally’s”). In the next example, the recognition hypotheses of two identical queries have only a single word in common:

Q1. *I get in the house Google.* (“I did it Google”)

Q2. *I did it crash cool.* (“I did it Google”)

Conversely, recognition hypotheses that are nearly identical are not necessarily retries. Often, these are “serial queries,” a series of queries the user is making of the same form or on the same topic, often to test the system.

Q1. *How tall is George Clooney?*

Q2. *How old is George Clooney?*

Q1. *Weather in New York.*

Q2. *Weather in Los Angeles.*

These complementary problems mean that we cannot use naïve text similarity features to identify retries. Instead, we combine features that model the first query’s likely accuracy to broader similarity features to form a more nuanced picture of a likely retry.

The five granular retry labels were collapsed into binary categories: search retry, other, and no retry were mapped to NO RETRY; and repetition and rephrase were mapped to RETRY. The label

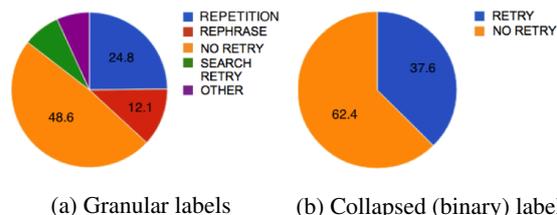


Figure 2: Retry label distribution.

distribution of the final dataset is shown in Figure 2.

4 Features

The features we consider can be divided into three main categories. The first group of features, *similarity*, is intended to measure the similarity between the two queries, as similar queries are (with the above caveats) more likely to be retries. We calculate the edit distance between the two transcripts at the character and word level, as well as the two most similar phonetic rewrites. We include both raw and normalized values as features. We also count the number of unigrams the two transcripts have in common and the length, absolute and relative, of the longest unigram overlap.

As we have shown in the previous section, similarity features alone cannot identify a retry, since ASR errors and user rephrases can result in recognition hypotheses that are significantly different from the original query, while a nearly identical pair of queries can have different search intents. Our second group of features, *correctness*, goes up a level in our labeling decision tree (Figure 1) and attempts to instead answer the question: “Was the first query transcribed incorrectly?” We use the confidence score assigned by the recognizer to the first recognition hypothesis as a measure of the system’s opinion of its own performance. Since this score, while informative, may be inaccurate, we also consider signals from the user that might indicate the accuracy of the hypothesis. A boolean feature indicates whether the user interacted with any of the results (structured or unstructured) that were presented by the system in response to the first query, which should constitute an implicit acceptance of the system’s recognition hypothesis. The length of the interval between the two queries is another feature, since a query that occurs immediately after another is likely to be a retry. We also include the difference and ratio of the two queries’ speaking rate, roughly calculated as the number of vowels divided by the audio duration in sec-

onds, since a speaker is likely to hyperarticulate (speak more loudly and slowly) after being misunderstood ((Wade et al., 1992; Oviatt et al., 1996; Levow, 1998; Bell and Gustafson, 1999; Soltau and Waibel, 1998)).

The third feature group, *recognizability*, attempts to model the characteristics of a query that is likely to be misrecognized (for the first query of the pair) or is likely to be a retry of a previous query (for the second query). We look at the language model (LM) score and the number of alternate pronunciations of the first query, predicting that a misrecognized query will have a lower LM score and more alternate pronunciations. In addition, we look at the number of characters and unigrams and the audio duration of each query, with the intuition that the length of a query may be correlated with its likelihood of being retried (or a retry). This feature group also includes two heuristic features intended to flag the “serial queries” mentioned before: the number of capitalized words in each query, and whether each one begins with a question word (who, what, etc.).

5 Prediction task

5.1 Experimental Results

A logistic regression model was trained on these features to predict the collapsed binary categories of NO RETRY (search retry, other, no retry) vs. RETRY (rephrase, repetition). The results of running this model with each combination of the feature groups are shown in Table 1.

Features	Precision	Recall	F1	Accuracy
Similarity	0.54	0.65	0.59	0.72
Correctness	0.53	0.67	0.59	0.73
Recognizability	0.49	0.63	0.55	0.70
Sim. & Corr.	0.67	0.71	0.69	0.77
Sim. & Rec.	0.62	0.70	0.66	0.76
Corr. & Rec.	0.65	0.71	0.68	0.77
All Features	0.70	0.76	0.73	0.81

Table 1: Results of the binary prediction task.

Individually, each feature group performed significantly better than the baseline strategy of always predicting NO RETRY (62.4%). Each pair of feature groups performed better than any individual group, and the final combination of all three feature groups had the highest precision, recall, and accuracy, suggesting that each aspect of the retry conceptualization provides valuable information to the model.

Of the *similarity* features, the ones that contributed significantly in the final model were character edit distance (normalized) and phoneme edit distance (raw and normalized); as expected, retries are associated with more similar query pairs. Of the *correctness* features, high recognizer confidence, the presence of a positive reaction from the user such as a link click, and a long interval between queries were all negatively associated with retries. The significant *recognizability* features included length of the first query in characters (longer queries were less likely to be retried) and the number of capital letters in each query (as our LM is case-sensitive): queries transcribed with more capital letters were more likely to be retried, but less likely to themselves be retries. In addition, the language model likelihood for the first query was, as expected, significantly lower for retries. Interestingly, the score of the *second* query was lower for retries as well. This accords with our finding that retries of misrecognized queries are themselves misrecognized 60%-70% of the time, which highlights the potential value of corrective action informed by the retry context.

Several features, though not significant in the model, are significantly different between the RETRY and NO RETRY categories, which affords us further insight into the characteristics of a retry. *T*-tests between the two categories showed that all edit distance features—character, word, reduced, and phonetic; raw and normalized—are significantly more similar between retry query pairs.¹ Similarly, the number of unigrams the two queries have in common is significantly higher for retries. The duration of each member of the query pair, in seconds and word count, is significantly more similar between retry pairs, and each member of a retry pair tends to be shorter than members of a no retry pair. Finally, members of NO RETRY query pairs were significantly more similar in speaking rate, and the relative speaking rate of the second query was significantly *slower* for RETRY pairs, possibly due to hyperarticulation.

5.2 Analysis

Figure 3 shows a breakdown of the true granular labels versus the predicted binary labels. The primary source of error is the REPHRASE category, which is identified as a retry with only 16.5% ac-

¹*T*-tests reported here use a conservative significance threshold of $p < 0.00125$ to control for family-wise type I error (“data dredging” effects).

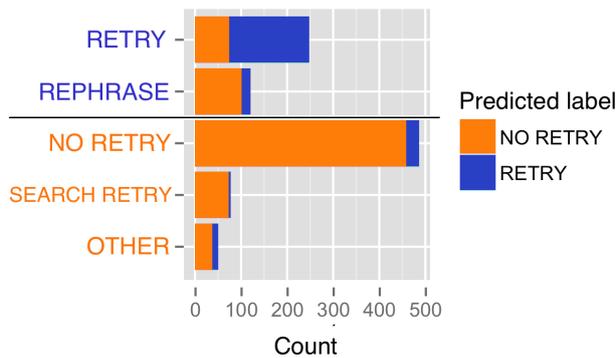


Figure 3: Performance on each of the granular categories.

curacy. This result reflects the fact that although rephrases conceptually belong in the retry category, their characteristics are materially different. Most notably, all edit distance features are significantly greater for rephrases. Differences in duration between the two queries in a pair, in seconds and words, are significantly greater as well. Rephrases also are significantly longer, in seconds and words, than strict retries. The model including only correctness and recognizability features does significantly better on rephrases than the full model, identifying them as retries with 25.6% accuracy, confirming that the similarity features are the primary culprit. Future work may address this issue by including features crafted to examine the similarity between substrings of the two queries, rather than the query as a whole, and by expanding the similarity definition to include synonyms.

To test the model’s performance with a larger, unseen dataset, we looked at how many retries it detected in the test set of potential retry pairs ($n=7,254$). We do not have retry annotations for this larger set, but we have transcriptions for the first member of each query pair, enabling us to calculate the word error rate (WER) of each query’s recognition hypothesis, and thus obtain ground truth for half of our retry definition. A perfect model should never predict RETRY when the first query is transcribed correctly ($WER==0$). As shown in Figure 4, our model assigns a RETRY label to approximately 14% of the queries following an incorrectly recognized search, and only 2% of queries following a correctly recognized search. While this provides us with only a lower bound on our model’s error, this significant correlation with an orthogonal accuracy metric shows that we have modeled at least this aspect of retries correctly, and suggests a correlation between retry rate and traditional WER-based evaluation.

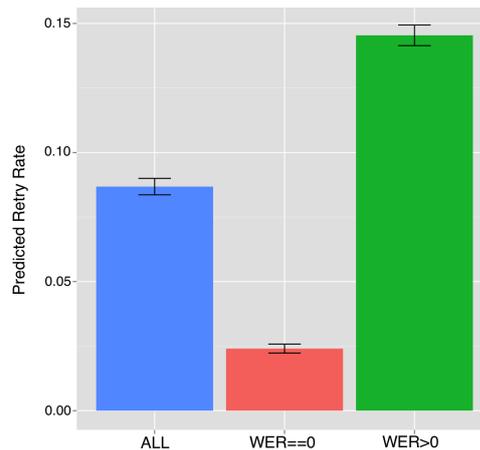


Figure 4: Performance on unseen data. A perfect model would have a predicted retry rate of 0 when $WER==0$.

6 Conclusion

We have presented a method for characterizing retries in an unrestricted voice interface to a search system. One particular challenge is the lack of simplifying assumptions based on domain and state (as users may consider the system to be stateless when issuing subsequent queries). We introduce a labeling scheme for retries that encompasses rephrases—cases in which the user reworded her query to adapt to the system’s error—as well as repetitions.

Our model identifies retries with 81% accuracy, significantly above baseline. Our error analysis confirms that user rephrasings complicate the binary class separation; an approach that models typical *typed* rephrasings may help overcome this difficulty. However, our model’s performance today correlates strongly with an orthogonal accuracy metric, word error rate, on unseen data. This suggests that “retry rate” is a reasonable offline quality metric, to be considered in context among other metrics and traditional evaluation based on word error rate.

Acknowledgments

The authors thank Daisy Stanton and Maryam Kamvar for their helpful comments on this project.

References

- Linda Bell and Joakim Gustafson. 1999. Repetition and its phonetic realizations: Investigating a swedish database of spontaneous computer-directed speech. In *Proceedings of ICPhS*, volume 99, pages 1221–1224.
- Mert Cevik, Fuliang Weng, and Chin-Hui Lee. 2008. Detection of repetitions in spontaneous speech in di-

- ologue sessions. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association (INTERSPEECH 2008)*, pages 471–474, Brisbane, Australia.
- Gina-Anne Levow. 1998. Characterizing and recognizing spoken corrections in human-computer dialogue. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 736–742. Association for Computational Linguistics.
- Marco Orlandi, Christopher Culy, and Horacio Franco. 2003. Using dialog corrections to improve speech recognition. In *Error Handling in Spoken Language Dialogue Systems*. International Speech Communication Association.
- Sharon Oviatt, G-A Levow, Margaret MacEachern, and Karen Kuhn. 1996. Modeling hyperarticulate speech during human-computer error resolution. In *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, volume 2, pages 801–804. IEEE.
- Hagen Soltau and Alex Waibel. 1998. On the influence of hyperarticulated speech on recognition performance. In *ICSLP*. Citeseer.
- Elizabeth Wade, Elizabeth Shriberg, and Patti Price. 1992. User behaviors affecting speech recognition. In *ICSLP*.
- Jason D. Williams. 2008. Exploiting the asr n-best by tracking multiple dialog state hypotheses. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association (INTERSPEECH 2008)*, pages 191–194, Brisbane, Australia.
- Geoffrey Zweig, Dan Bohus, Xiao Li, and Patrick Nguyen. 2008. Structured models for joint decoding of repeated utterances. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association (INTERSPEECH 2008)*, pages 1157–1160, Brisbane, Australia.
- Geoffrey Zweig. 2009. New methods for the analysis of repeated utterances. In *Proceedings of the 10th Annual Conference of the International Speech Communication Association (INTERSPEECH 2009)*, pages 2791–2794, Brighton, United Kingdom.

Sliding Alignment Windows for Real-Time Crowd Captioning

Mohammad Kazemi, Rahman Lavaee, Iftekhar Naim and Daniel Gildea

Dept. of Electrical and Computer Engineering and

Dept. of Computer Science

University of Rochester

Rochester, NY 14627

Abstract

The primary way of providing real-time speech to text captioning for hard of hearing people is to employ expensive professional stenographers who can type as fast as natural speaking rates. Recent work has shown that a feasible alternative is to combine the partial captions of ordinary typists, each of whom is able to type only part of what they hear. In this paper, we extend the state of the art fixed-window alignment algorithm (Naim et al., 2013) for combining the individual captions into a final output sequence. Our method performs alignment on a sliding window of the input sequences, drastically reducing both the number of errors and the latency of the system to the end user over the previously published approaches.

1 Introduction

Real-time captioning provides deaf or hard of hearing people access to speech in mainstream classrooms, at public events, and on live television. Studies performed in the classroom setting show that the latency between when a word was said and when it is displayed must be under five seconds to maintain consistency between the captions being read and other visual cues (Wald, 2005; Kushalnagar et al., 2014). The most common approach to real-time captioning is to recruit a trained stenographer with a special purpose phonetic keyboard, who transcribes the speech to text with less than five seconds of latency. Unfortunately, professional captionists are quite expensive (\$150 per hour), must be recruited in blocks of an hour or more, and are difficult to schedule on short

Merging Incomplete Captions

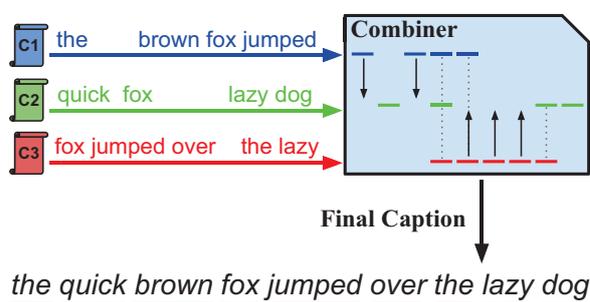


Figure 1: General layout of crowd captioning systems. Captionists (C1, C2, C3) submit partial captions that are automatically combined into a high-quality output.

notice. Automatic speech recognition (ASR) systems (Saraclar et al., 2002), on the other hand, attempts to provide a cheap and fully automated solution to this problem. However, the accuracy of ASR quickly plummets to below 30% when used on an untrained speaker's voice, in a new environment, or in the absence of a high quality microphone (Wald, 2006). The accuracy of the ASR systems can be improved using the 're-speaking' technique, which requires a person that the ASR has been trained on to repeat the words said by a speaker as he hears them. Simultaneously hearing and speaking, however, is not straightforward, and requires some training.

An alternative approach is to combine the efforts of multiple non-expert captionists (anyone who can type), instead of relying on trained workers (Lasecki et al., 2012; Naim et al., 2013). In this approach, multiple non-expert human workers transcribe an audio stream containing speech in real-time. Workers type as much as they can of

the input, and, while no one worker’s transcript is complete, the portions captured by various workers tend to overlap. For each input word, a timestamp is recorded, indicating when the word is typed by a worker. The partial inputs are combined to produce a final transcript (see Figure 1). This approach has been shown to dramatically outperform ASR in terms of both accuracy and Word Error Rate (WER) (Lasecki et al., 2012; Naim et al., 2013). Furthermore, recall of individual words irrespective of their order approached and even exceeded that of a trained expert stenographer with seven workers contributing, suggesting that the information is present to meet the performance of a stenographer (Lasecki et al., 2012). However, aligning these individual words in the correct sequential order remains a challenging problem.

Lasecki et al. (2012) addressed this alignment problem using off-the-shelf multiple sequence alignment tools, as well as an algorithm based on incrementally building a precedence graph over output words. Improved results for the alignment problem were shown using weighted A* search by Naim et al. (2013). To speed the search for the best alignment, Naim et al. (2013) divided sequences into chunks of a fixed time duration, and applied the A* alignment algorithm to each chunk independently. Although this method speeds the search for the best alignment, it introduces a significant number of errors to the output of the system due to inconsistency at the boundaries of the chunks. In this paper, we introduce a novel sliding window technique which avoids the errors produced by previous systems at the boundaries of the chunks used for alignment. This technique produces dramatically fewer errors for the same amount of computation time.

2 Problem Overview and Background

The problem of aligning and combining multiple transcripts can be mapped to the well-studied Multiple Sequence Alignment (MSA) problem (Edgar and Batzoglou, 2006). Let $S_1, \dots, S_K, K \geq 2$, be the K sequences over an alphabet Σ , and having length N_1, \dots, N_K . For the caption alignment task, we treat each individual word as a symbol in our alphabet Σ . The special gap symbol ‘-’ represents a missing word and does not belong to Σ . Let $A = (a_{ij})$ be a $K \times N_f$ matrix, where $a_{ij} \in \Sigma \cup \{-\}$, and the i^{th} row has exactly $(N_f - N_i)$ gaps and is identical to S_i if we ignore

Algorithm 1 MSA-A* Algorithm

Require: K input sequences $\mathcal{S} = \{S_1, \dots, S_K\}$ having length N_1, \dots, N_K , heuristic weight w , beam size b
input $start \in \mathbb{N}^K, goal \in \mathbb{N}^k$
output an $N \times K$ matrix of integers indicating the index into each input sequence of each position in the output sequence

- 1: $g(start) \leftarrow 0, f(start) \leftarrow w \times h(start)$.
- 2: $Q \leftarrow \{start\}$
- 3: **while** $Q \neq \emptyset$ **do**
- 4: $n \leftarrow \text{EXTRACT-MIN}(Q)$
- 5: **for all** $s \in \{0, 1\}^K - \{0^K\}$ **do**
- 6: $n_i \leftarrow n + s$
- 7: **if** $n_i = goal$ **then**
- 8: Return the alignment matrix for the reconstructed path from $start$ to n_i
- 9: **else if** $n_i \notin \text{Beam}(b)$ **then**
- 10: continue;
- 11: **else**
- 12: $g(n_i) \leftarrow g(n) + c(n, n_i)$
- 13: $f(n_i) \leftarrow g(n_i) + w \times h(n_i)$
- 14: INSERT-ITEM($Q, n_i, f(n_i)$)
- 15: **end if**
- 16: **end for**
- 17: **end while**

the gaps. Every column of A must have at least one non-gap symbol. Therefore, the j^{th} column of A indicates an alignment state for the j^{th} position, where the state can have one of the $2^K - 1$ possible combinations. Our goal is to find the optimum alignment matrix A_{OPT} that minimizes the sum of pairs (SOP) cost function:

$$c(A) = \sum_{1 \leq i \leq j \leq K} c(A_{ij}) \quad (1)$$

where $c(A_{ij})$ is the cost of the pairwise alignment between S_i and S_j according to A . Formally, $c(A_{ij}) = \sum_{l=1}^{N_f} \text{sub}(a_{il}, a_{jl})$, where $\text{sub}(a_{il}, a_{jl})$ denotes the cost of substituting a_{jl} for a_{il} . If a_{il} and a_{jl} are identical, the substitution cost is zero. The substitution cost for two words is estimated based on the edit distance between two words. The exact solution to the SOP optimization problem is NP-Complete (Wang and Jiang, 1994), but many methods solve it approximately. Our approach is based on weighted A* search for approximately solving the MSA problem (Lermen and Reinert, 2000; Naim et al., 2013).

2.1 Weighted A* Search for MSA

The problem of minimizing the SOP cost function for K sequences is equivalent to estimating the shortest path between a single source node and a single sink node in a K -dimensional mesh graph, where each node corresponds to a distinct position in the K sequences. The source node is $[0, \dots, 0]$

Algorithm 2 Fixed Window Algorithm

Require: K input sequences $\mathcal{S} = \{S_1, \dots, S_K\}$ having length N_1, \dots, N_K , window parameter $chunk_length$.

- 1: $start_time \leftarrow 0$
- 2: **while** $goal \prec [N_1, \dots, N_K]$ **do**
- 3: **for all** i **do**
- 4: $start[i] \leftarrow closest_word(i, start_time)$
- 5: **end for**
- 6: $end_time \leftarrow start_time + chunk_length$
- 7: **for all** i **do**
- 8: $goal[i] \leftarrow closest_word(i, end_time) - 1$
- 9: **end for**
- 10: $alignmatrix \leftarrow MSA-A^*(start, goal)$
- 11: concatenate $alignmatrix$ onto end of $finalmatrix$
- 12: $start_time \leftarrow end_time$
- 13: **end while**
- 14: Return $finalmatrix$

and the sink node is $[N_1, \dots, N_K]$. The total number of nodes in the lattice is $(N_1 + 1) \times (N_2 + 1) \times \dots \times (N_K + 1)$, and each node has $2^K - 1$ possible successors and predecessors. The A* search algorithm treats each node position $n = [n_1, \dots, n_K]$ as a search state, and estimates the cost function $g(n)$ and the heuristic function $h(n)$ for each state. The cost function $g(n)$ represents the exact minimum SOP cost to align the K sequences from the beginning to the current position. The heuristic function represents the approximate minimum cost of aligning the suffixes of the K sequences, starting after the current position n . The commonly used heuristic function is $h_{pair}(n)$:

$$h_{pair}(n) = L(n \rightarrow t) = \sum_{1 \leq i < j \leq K} c(A_p^*(\sigma_i^n, \sigma_j^n)) \quad (2)$$

where $L(n \rightarrow t)$ denotes the lower bound on the cost of the shortest path from n to destination t , A_p^* is the optimal pairwise alignment, and σ_i^n is the suffix of node n in the i -th sequence. The weighted A* search uses a priority queue Q to store the search states n . At each step of the A* search algorithm, the node with the smallest evaluation function, $f(n) = g(n) + wh_{pair}(n)$ (where $w \geq 1$), is extracted from the priority queue Q and expanded by one edge. The search continues until the goal node is extracted from Q . To further speed up the search, a beam constraint is applied on the search space using the timestamps of each individual input words. If the beam size is set to b seconds, then any state that aligns two words having more than b seconds time lag is ignored. The detailed procedure is shown in Algorithm 1. After the alignment, the captions are combined via majority voting at each position of the alignment

matrix. We ignore the alignment columns where the majority vote is below a certain threshold t_v (typically $t_v = 2$), and thus filter out spurious errors and spelling mistakes.

Although weighted A* significantly speeds the search for the best alignment, it is still too slow for very long sequences. For this reason, Naim et al. (2013) divided the sequences into chunks of a fixed time duration, and applied the A* alignment algorithm to each chunk independently. The chunks were concatenated to produce the final output sequence, as shown in Algorithm 2.

2.2 Limitations of Fixed Window Algorithm

The fixed window based alignment has two key limitations. First, aligning disjoint chunks independently tends to introduce a large number of errors at the boundary of each chunk. This is because the chunk boundaries are defined with respect to the timestamps associated with each word in the captions, but the timestamps can vary greatly between words that should in fact be aligned. After all, if the timestamps corresponded precisely to the original time at which each word was spoken, the entire alignment problem would be trivial. The fact that the various instances of a single word in each transcription may fall on either side of a chunk boundary leads to errors where a word is either duplicated in the final output for more than one chunk, or omitted entirely. This problem also causes errors in ordering among the words remaining within one chunk, because there is less information available to constrain the ordering relations between transcriptions. Second, the fixed window alignment algorithm requires longer chunks (≥ 10 seconds) to obtain reasonable accuracy, and thus introduces unsatisfactory latency.

3 Sliding Alignment Windows

In order to address the problems described above, we explore a technique based on a sliding alignment window, shown in Algorithm 3. We start with alignment with a fixed chunk size. After aligning the first chunk, we use the information derived from the alignment to determine where the next chunk should begin within each transcription. We use a single point in the aligned output as the starting point for the next chunk, and determine the corresponding starting position within each original transcription. This single point is determined by a tunable parameter $keep_length$

Algorithm 3 Sliding Window Algorithm

Require: K input sequences $\mathcal{S} = \{S_1, \dots, S_K\}$ having length N_1, \dots, N_K , window parameters $chunk_length$ and $keep_length$.

```

1:  $start \leftarrow 0^K, goal \leftarrow 0^K$ 
2: while  $goal \prec [N_1, \dots, N_K]$  do
3:    $endtime \leftarrow chunk\_length + \max_i time(start[i])$ 
4:   for all  $i$  do
5:      $goal[i] \leftarrow closest\_word(i, endtime)$ 
6:   end for
7:    $alignmatrix \leftarrow MSA-A^*(start, goal)$ 
8:   concatenate first  $keep\_length$  columns of  $alignmatrix$  onto end of  $finalmatrix$ 
9:   for all  $i$  do
10:     $start[i] \leftarrow alignmatrix[keep\_length][i]$ 
11:   end for
12: end while
13: Return  $finalmatrix$ 

```

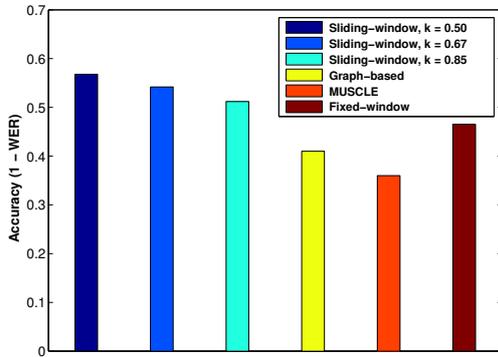


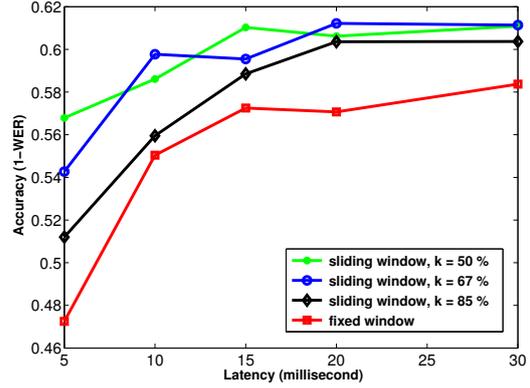
Figure 2: Evaluation of different systems on using WER metric for measuring transcription quality.

(line 10 of Algorithm 3). The materials in the output alignment that follow this point is thrown away, and replaced with the output produced by aligning the next chunk starting from this point (line 8). The process continues iteratively, allowing us to avoid using the erroneous output alignments in the neighborhood of the arbitrary endpoints for each chunk.

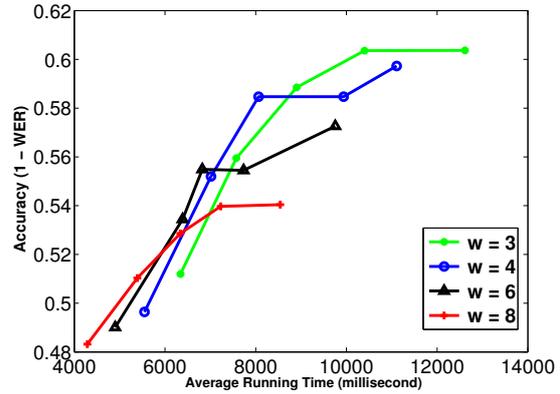
4 Experimental Results

We evaluate our system on a dataset of four 5-minute long audio clips of lectures in electrical engineering and chemistry lectures taken from MIT OpenCourseWare. The same dataset used by (Lasecki et al., 2012) and (Naim et al., 2013). Each audio clip is transcribed by 10 non-expert human workers in real time. We measure the accuracy in terms of Word Error Rate (WER) with respect to a reference transcription.

We are interested in investigating how the three



(a) varying keep-lengths for fixed heuristic weight



(b) varying heuristic weights for fixed keep-length

Figure 3: Tradeoff between speed and accuracy for different heuristic weights and keep-lengths

key parameters of the algorithm, i.e., the chunk size (c), the heuristic weight (w) and the keep-length (k), affect the system latency, the search speed, and the alignment accuracy. The chunk size directly determines the latency of the system to the end user, as alignment cannot begin until an entire chunk is captured. Furthermore, the chunk size, the heuristic weight, and the keep-length help us to trade-off speed versus accuracy. We also compare the performance of our algorithm with that of the most accurate fixed alignment window algorithm (Naim et al., 2013). The performance in terms of WER for sliding and fixed alignment windows is presented in Figure 2. Out of the systems in Figure 2, the first three systems consist of sliding alignment window algorithm with different values of keep-length parameter: (1) keep-length = 0.5; (2) keep-length = 0.67; and (3) keep-length = 0.85. The other systems are the graph-based algorithm of (Lasecki et al., 2012), the MUSCLE algorithm of (Edgar, 2004), and the most accu-

rate fixed alignment window algorithm of (Naim et al., 2013). We set the heuristic weight parameter (w) to 3 and the chunk size parameter (c) to 5 seconds for all the three sliding window systems and the fixed window system. Sliding alignment window produces better results and outperforms the other algorithms even for large values of the keep-length parameter. The sliding alignment window with keep-length 0.5 achieves 0.5679 average accuracy in terms of (1-WER), providing a 18.09% improvement with respect to the most accurate fixed alignment window (average accuracy 0.4857). On the same dataset, Lasecki et al. (2012) reported 36.6% accuracy using the Dragon Naturally Speaking ASR system (version 11.5 for Windows).

To show the trade-off between latency and accuracy, we fix the heuristic weight ($w = 3$) and plot the accuracy as a function of chunk size in Figure 3. We repeat this experiment for different values of keep-length. We observe that the sliding window approach dominates the fixed window approach across a wide range of chunk sizes. Furthermore, we can see that for smaller values of the chunk size parameter, increasing the keep-length makes the system less accurate. As the chunk size parameter increases, the performance of sliding window systems with different values of keep-length parameter converges. Therefore, at larger chunk sizes, for which there are smaller number of boundaries, the keep-length parameter has lower impact.

Next, we show the trade-off between computation speed and accuracy in Figure 3, as we fix the heuristic weight and vary the chunk size over the range [5, 10, 15, 20, 30] seconds. Larger chunks are more accurately aligned but require computation time that grows as N^K in the chunk size N in the worst case. Furthermore, smaller weights allow faster alignment, but provide lower accuracy.

5 Conclusion

In this paper, we present a novel sliding window based text alignment algorithm for real-time crowd captioning. By effectively addressing the problem of alignment errors at chunk boundaries, our sliding window approach outperforms the existing fixed window based system (Naim et al., 2013) in terms of word error rate, particularly when the chunk size is small, and thus achieves higher accuracy at lower latency.

Acknowledgments Funded by NSF awards IIS-1218209 and IIS-0910611.

References

- Robert C Edgar and Serafim Batzoglou. 2006. Multiple sequence alignment. *Current opinion in structural biology*, 16(3):368–373.
- Robert C Edgar. 2004. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792–1797.
- Raja S Kushalnagar, Walter S Lasecki, and Jeffrey P Bigham. 2014. Accessibility evaluation of classroom captions. *ACM Transactions on Accessible Computing (TACCESS)*, 5(3):7.
- Walter Lasecki, Christopher Miller, Adam Sadilek, Andrew Abumoussa, Donato Borrello, Raja Kushalnagar, and Jeffrey Bigham. 2012. Real-time captioning by groups of non-experts. In *Proceedings of the 25rd annual ACM symposium on User interface software and technology, UIST '12*.
- Martin Lermen and Knut Reinert. 2000. The practical use of the A* algorithm for exact multiple sequence alignment. *Journal of Computational Biology*, 7(5):655–671.
- Iftekhar Naim, Daniel Gildea, Walter Lasecki, and Jeffrey Bigham. 2013. Text alignment for real-time crowd captioning. In *Proceedings of the 2013 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL-13)*.
- Murat Saraclar, Michael Riley, Enrico Bocchieri, and Vincent Goffin. 2002. Towards automatic closed captioning: Low latency real time broadcast news transcription. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, pages 1741–1744.
- Mike Wald. 2005. Using automatic speech recognition to enhance education for all students: Turning a vision into reality. In *Proceedings 35th Annual Conference on Frontiers in Education, 2005. FIE '05.*, pages S3G–S3G, Oct.
- Mike Wald. 2006. Creating accessible educational multimedia through editing automatic speech recognition captioning in real time. *Interactive Technology and Smart Education*, 3(2):131–141.
- Lusheng Wang and Tao Jiang. 1994. On the complexity of multiple sequence alignment. *Journal of Computational Biology*, 1(4):337–348.

Detection of Topic and its Extrinsic Evaluation Through Multi-Document Summarization

Yoshimi Suzuki

Interdisciplinary Graduate School of
Medicine and Engineering
University of Yamanashi
Kofu, 400-8511, JAPAN
ysuzuki@yamanashi.ac.jp

Fumiyo Fukumoto

Interdisciplinary Graduate School of
Medicine and Engineering
University of Yamanashi
Kofu, 400-8511, JAPAN
fukumoto@yamanashi.ac.jp

Abstract

This paper presents a method for detecting words related to a topic (we call them topic words) over time in the stream of documents. Topic words are widely distributed in the stream of documents, and sometimes they frequently appear in the documents, and sometimes not. We propose a method to reinforce topic words with low frequencies by collecting documents from the corpus, and applied Latent Dirichlet Allocation (Blei et al., 2003) to these documents. For the results of LDA, we identified topic words by using Moving Average Convergence Divergence. In order to evaluate the method, we applied the results of topic detection to extractive multi-document summarization. The results showed that the method was effective for sentence selection in summarization.

1 Introduction

As the volume of online documents has drastically increased, the analysis of topic bursts, topic drift or detection of topic is a practical problem attracting more and more attention (Allan et al., 1998; Swan and Allan, 2000; Allan, 2003; Klinkenberg, 2004; Lazarescu et al., 2004; Folino et al., 2007). The earliest known approach is the work of Klinkenberg and Joachims (Klinkenberg and Joachims, 2000). They have attempted to handle concept changes by focusing a window with documents sufficiently close to the target concept. Mane *et. al.* proposed a method to generate maps that support the identification of major research topics and trends (Mane and Borner, 2004). The method used Kleinberg's burst detection algorithm, co-occurrences of words, and graph layout technique. Scholz *et. al.* have attempted to use different ensembles obtained by training several data streams to detect concept drift (Scholz,

2007). However the ensemble method itself remains a problem that how to manage several classifiers effectively. He and Parket attempted to find bursts, periods of elevated occurrence of events as a dynamic phenomenon instead of focusing on arrival rates (He and Parker, 2010). However, the fact that topics are widely distributed in the stream of documents, and sometimes they frequently appear in the documents, and sometimes not often hamper such attempts.

This paper proposes a method for detecting topic over time in series of documents. We reinforced words related to a topic with low frequencies by collecting documents from the corpus, and applied Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to these documents in order to extract topic candidates. For the results of LDA, we applied Moving Average Convergence Divergence (MACD) to find topic words while He *et. al.*, applied it to find bursts. The MACD is a technique to analyze stock market trends (Murphy, 1999). It shows the relationship between two moving averages of prices modeling bursts as intervals of topic dynamics, *i.e.*, positive acceleration. Fukumoto *et. al.* also applied MACD to find topics. However, they applied it only to the words with high frequencies in the documents (Fukumoto et al., 2013). In contrast, we applied it to the topic candidates obtained by LDA.

We examined our method by extrinsic evaluation, *i.e.*, we applied the results of topic detection to extractive multi-document summarization. We assume that a salient sentence includes words related to the target topic, and an event of each documents. Here, an event is something that occurs at a specific place and time associated with some specific actions (Allan et al., 1998). We identified event words by using the traditional *tf*idf* method applied to the results of named entities. Each sentence in documents is represented using a vector of frequency weighted words that can be event

or topic words. We used Markov Random Walk (MRW) to compute the rank scores for the sentences (Page et al., 1998). Finally, we selected a certain number of sentences according to the rank score into a summary.

2 Topic Detection

2.1 Extraction of Topic Candidates

LDA presented by (Blei et al., 2003) models each document as a mixture of topics (we call it `lda_topic` to discriminate our *topic* candidates), and generates a discrete probability distribution over words for each `lda_topic`. The generative process for LDA can be described as follows:

1. For each topic $k = 1, \dots, K$, generate ϕ_k , multinomial distribution of words specific to the topic k from a Dirichlet distribution with parameter β ;
2. For each document $d = 1, \dots, D$, generate θ_d , multinomial distribution of topics specific to the document d from a Dirichlet distribution with parameter α ;
3. For each word $n = 1, \dots, N_d$ in document d ;
 - (a) Generate a topic z_{dn} of the n^{th} word in the document d from the multinomial distribution θ_d
 - (b) Generate a word w_{dn} , the word associated with the n^{th} word in document d from multinomial $\phi_{z_{dn}}$

Like much previous work on LDA, we used Gibbs sampling to estimate ϕ and θ . The sampling probability for topic z_i in document d is given by:

$$P(z_i | z_{\setminus i}, W) = \frac{(n_{\setminus i, j}^v + \beta)(n_{\setminus i, j}^d + \alpha)}{(n_{\setminus i, j}^v + W\beta)(n_{\setminus i, \cdot}^d + T\alpha)}. \quad (1)$$

$z_{\setminus i}$ refers to a topic set Z , not including the current assignment z_i . $n_{\setminus i, j}^v$ is the count of word v in topic j that does not include the current assignment z_i , and $n_{\setminus i, j}^d$ indicates a summation over that dimension. W refers to a set of documents, and T denotes the total number of unique topics. After a sufficient number of sampling iterations, the approximated posterior can be used to estimate ϕ and θ by examining the counts of word assignments to topics and topic occurrences in documents. The

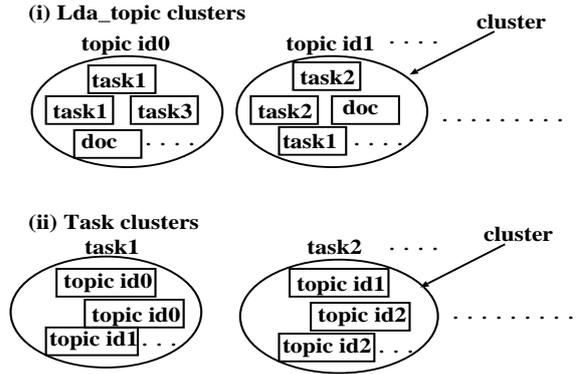


Figure 1: Lda_topic cluster and task cluster

approximated probability of topic k in the document d , $\hat{\theta}_d^k$, and the assignments word w to topic k , $\hat{\phi}_k^w$ are given by:

$$\hat{\theta}_d^k = \frac{N_{dk} + \alpha}{N_d + \alpha K}. \quad (2)$$

$$\hat{\phi}_k^w = \frac{N_{kw} + \beta}{N_k + \beta V}. \quad (3)$$

We used documents prepared by summarization tasks, NTCIR and DUC data as each task consists of series of documents with the same topic. We applied LDA to the set consisting of all documents in the summarization tasks and documents from the corpus. We need to estimate the appropriate number of `lda_topic`.

Let k' be the number of `lda_topics` and d' be the number of topmost d' documents assigned to each `lda_topic`. We note that the result obtained by LDA can be regarded as the two types of clustering result shown in Figure 1: (i) each cluster corresponds to each `lda_topic` (topic id0, topic id1 \dots in Figure 1), and each element of the clusters is the document in the summarization tasks (task1, task2, \dots in Figure 1) or from the corpus (doc in Figure 1), and (ii) each cluster corresponds to the summarization task and each element of the clusters is the document in the summarization tasks or the document from the corpus assigned topic id. For example, DUC2005 consists of 50 tasks. Therefore the number of different clusters is 50. We call the former `lda_topic` cluster and the latter task cluster. We estimated k' and d' by using Entropy measure given by:

$$E = -\frac{1}{\log l} \sum_j \frac{N_j}{N} \sum_i P(A_i, C_j) \log P(A_i, C_j) \quad (4)$$

l refers to the number of clusters. $P(A_i, C_j)$ is a probability that the elements of the cluster C_j assigned to the correct class A_i . N denotes the total number of elements and N_j shows the total number of elements assigned to the cluster C_j . The value of E ranges from 0 to 1, and the smaller value of E indicates better result. Let E_{topic} and E_{task} are entropy value of lda_topic cluster and task cluster, respectively. We chose the parameters k' and d' whose value of the summation of E_{topic} and E_{task} is smallest. For each lda_topic, we extracted words whose probabilities are larger than zero, and regarded these as topic candidates.

2.2 Topic Detection by MACD

The proposed method does not simply use MACD to find bursts, but instead determines topic words in series of documents. Unlike Dynamic Topic Models (Blei and Lafferty, 2006), it does not assume Gaussian distribution so that it is a natural way to analyze bursts which depend on the data. We applied it to extract topic words in series of documents. MACD histogram defined by Eq. (6) shows a difference between the MACD and its moving average. MACD of a variable x_t is defined by the difference of n_1 -day and n_2 -day moving averages, $MACD(n_1, n_2) = EMA(n_1) - EMA(n_2)$. Here, $EMA(n_i)$ refers to n_i -day Exponential Moving Average (EMA). For a variable $x = x(t)$ which has a corresponding discrete time series $\mathbf{x} = \{x_t \mid t = 0, 1, \dots\}$, the n -day EMA is defined by Eq. (5).

$$\begin{aligned} EMA(n)[x]_t &= \alpha x_t + (1 - \alpha)EMA(n-1)[x]_{t-1} \\ &= \sum_{k=0}^{n-1} \alpha(1 - \alpha)^k x_{t-k}. \end{aligned} \quad (5)$$

α refers to a smoothing factor and it is often taken to be $\frac{2}{n+1}$. MACD histogram shows a difference between the MACD and its moving average¹.

$$\text{hist}(n_1, n_2, n_3) = \frac{MACD(n_1, n_2) - EMA(n_3)[MACD(n_1, n_2)]}{EMA(n_3)[MACD(n_1, n_2)]}. \quad (6)$$

The procedure for topic detection with MACD is illustrated in Figure 2. Let A be a series of documents and w be one of the topic candidates obtained by LDA. Each document in A is sorted in chronological order. We set A to the documents from the summarization task. Whether or not a word w is a topic word is judged as follows:

¹In the experiment, we set n_1 , n_2 , and n_3 to 4, 8 and 5, respectively (He and Parker, 2010).

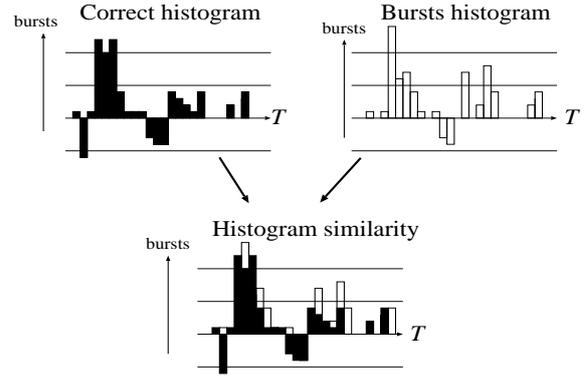


Figure 2: Topic detection with MACD

1. Create document-based MACD histogram where X-axis refers to T , *i.e.*, a period of time (numbered from day 1 to 365). Y-axis is the document count in A per day. Hereafter, referred to as correct histogram.
2. Create term-based MACD histogram where X-axis refers to T , and Y-axis denotes bursts of word w in A . Hereafter, referred to as bursts histogram.
3. We assume that if a term w is informative for summarizing a particular documents in a collection, its burstiness approximates the burstiness of documents in the collection. Because w is a representative word of each document in the task. Based on this assumption, we computed similarity between correct and word histograms by using KL-distance². Let P and Q be a normalized distance of correct histogram, and bursts histogram, respectively. KL-distance is defined by $D(P \parallel Q) = \sum_{i=1} P(x_i) \log \frac{P(x_i)}{Q(x_i)}$ where x_i refers bursts in time i . If the value of $D(P \parallel Q)$ is smaller than a certain threshold value, w is regarded as a topic word.

3 Extrinsic Evaluation to Summarization

3.1 Event detection

An event word is something that occurs at a specific place and time associated with some specific actions (Allan, 2003; Allan et al., 1998). It refers to notions of who(person), where(place),

²We tested KL-distance, histogram intersection and Bhattacharyya distance to obtain similarities. We reported only the result obtained by KL-distance as it was the best results among them.

when(time) including what, why and how in a document. Therefore, we can assume that named entities(NE) are linguistic features for event detection. An event word refers to the *theme* of the document itself, and frequently appears in the document but not frequently appear in other documents. Therefore, we first applied NE recognition to the target documents to be summarized, and then calculated tf*idf to the results of NE recognition. We extracted words whose tf*idf values are larger than a certain threshold value, and regarded these as event words.

3.2 Sentence extraction

We recall that our hypothesis about key sentences in multiple documents is that they include topic and event words. Each sentence in the documents is represented using a vector of frequency weighted words that can be event or topic words.

Like much previous work on extractive summarization (Erkan and Radev, 2004; Mihalcea and Tarau, 2005; Wan and Yang, 2008), we used Markov Random Walk (MRW) model to compute the rank scores for the sentences. Given a set of documents to be summarized, $G = (S, E)$ is a graph reflecting the relationships between two sentences. S is a set of vertices, and each vertex s_i in S is a sentence. E is a set of edges, and each edge e_{ij} in E is associated with an affinity weight $f(i \rightarrow j)$ between sentences s_i and s_j ($i \neq j$). The affinity weight is computed using cosine measure between the two sentences, s_i and s_j . Two vertices are connected if their affinity weight is larger than 0 and we let $f(i \rightarrow i) = 0$ to avoid self transition. The transition probability from s_i to s_j is then defined as follows:

$$p(i \rightarrow j) = \begin{cases} \frac{f(i \rightarrow j)}{\sum_{k=1}^{|S|} f(i \rightarrow k)}, & \text{if } \sum f \neq 0 \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

We used the row-normalized matrix $U_{ij} = (U_{ij})_{|S| \times |S|}$ to describe G with each entry corresponding to the transition probability, where $U_{ij} = p(i \rightarrow j)$. To make U a stochastic matrix, the rows with all zero elements are replaced by a smoothing vector with all elements set to $\frac{1}{|S|}$. The final transition matrix is given by formula (8), and each score of the sentence is obtained by the principal eigenvector of the matrix M .

$$M = \mu U^T + \frac{(1-\mu)}{|S|} \vec{e} \vec{e}^T. \quad (8)$$

We selected a certain number of sentences according to rank score into the summary.

4 Experiments

4.1 Experimental settings

We applied the results of topic detection to extractive multi-document summarization task, and examined how the results of topic detection affect the overall performance of the salient sentence selection. We used two tasks, Japanese and English summarization tasks, NTCIR-3³ SUMM Japanese and DUC⁴ English data. The baselines are (i) MRW model (**MRW**): The method applies the MRW model only to the sentences consisted of noun words, (ii) Event detection (**Event**): The method applies the MRW model to the result of event detection, (iii) Topic Detection by LDA (**LDA**): MRW is applied to the result of topic candidates detection by LDA and (iv) Topic Detection by LDA and MACD (**LDA & MACD**): MRW is applied to the result of topic detection by LDA and MACD only, *i.e.*, the method does not include event detection.

4.2 NTCIR data

The data used in the NTCIR-3 multi-document summarization task is selected from 1998 to 1999 of Mainichi Japanese Newspaper documents. The gold standard data provided to human judges consists of FBFREE DryRun and FormalRun. Each data consists of 30 tasks. There are two types of correct summary according to the character length, “long” and “short”, All series of documents were tagged by CaboCha (Kudo and Matsumoto, 2003). We used person name, organization, place and proper name extracted from NE recognition (Kudo and Matsumoto, 2003) for event detection, and noun words including named entities for topic detection. FBFREE DryRun data is used to tuning parameters, *i.e.*, the number of extracted words according to the tf*idf value, and the threshold value of KL-distance. The size that optimized the average Rouge-1(R-1) score across 30 tasks was chosen. As a result, we set tf*idf and KL-distance to 100 and 0.104, respectively.

We used FormalRun as a test data, and another set consisted of 218,724 documents from 1998 to 1999 of Mainichi newspaper as a corpus used in

³<http://research.nii.ac.jp/ntcir/>

⁴<http://duc.nist.gov/pubs.html>

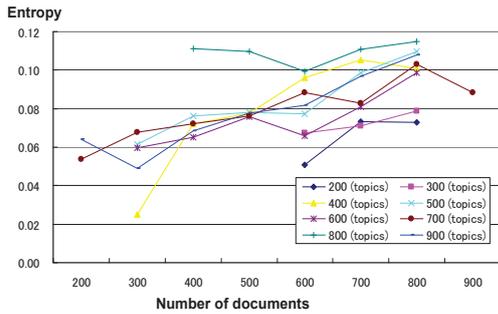


Figure 3: Entropy against the # of topics and documents

Method	Short	Long
	R-1	R-1
MRW	.369	.454
Event	.625	.724
LDA	.525	.712
LDA & MACD	.630	.742
Event & Topic	.678	.744

Table 1: Sentence Extraction (NTCIR-3 test data)

LDA and MACD. We estimated the number of k' and d' in LDA, *i.e.*, we searched k' and d' in steps of 100 from 200 to 900. Figure 3 illustrates entropy value against the number of topics k' and documents d' using 30 tasks of FormalRun data. Each plot shows that at least one of the documents for each summarization task is included in the cluster. We can see from Figure 3 that the value of entropy depends on the number of documents rather than the number of topics. From the result shown in Figure 3, the minimum entropy value was 0.025 and the number of topics and documents were 400 and 300, respectively. We used them in the experiment. The summarization results are shown in Table 1.

Table 1 shows that our approach, “Event & Topic” outperforms other baselines, regardless of the summary type (long/short). Topic candidates include surplus words that are not related to the topic because the results obtained by “LDA” were worse than those obtained by “LDA & MACD”, and even worse than “Event” in both short and long summary. This shows that integration of LDA and MACD is effective for topic detection.

4.3 DUC data

We used DUC2005 consisted of 50 tasks for training, and 50 tasks of DUC2006 data for testing in order to estimate parameters. We set $tf \cdot idf$ and

Method	R-1	Method	R-1
MRW	.381	Event	.407
LDA	.402	LDA & MACD	.428
Event & Topic	.438		
PYTHY	.426	HybHSum	.456
hPAM	.412	TTM	.447

Table 2: Comparative results (DUC2007 test data)

KL-distance to 80 and 0.9. The minimum entropy value was 0.050 and the number of topics and documents were 500 and 600, respectively. 45 tasks from DUC2007 were used to evaluate the performance of the method. All documents were tagged by Tree Tagger (Schmid, 1995) and Stanford Named Entity Tagger⁵ (Finkel et al., 2005). We used person name, organization and location for event detection, and noun words including named entities for topic detection. AQUAINT corpus⁶ which consists of 1,033,461 documents are used as a corpus in LDA and MACD. Table 2 shows Rouge-1 against unigrams.

We can see from Table 2 that Rouge-1 obtained by our approach was also the best compared to the baselines. Table 2 also shows the performance of other research sites reported by (Celikylmaz and Hakkani-Tur, 2010). The top site was “HybHSum” by (Celikylmaz and Hakkani-Tur, 2010). However, the method is a semi-supervised technique that needs a tagged training data. Our approach achieves performance approaching the top-performing unsupervised method, “TTM” (Celikylmaz and Hakkani-Tur, 2011), and is competitive to “PYTHY” (Toutanova et al., 2007) and “hPAM” (Li and McCallum, 2006). Prior work including “TTM” has demonstrated the usefulness of semantic concepts for extracting salient sentences. For future work, we should be able to obtain further advantages in efficacy in our topic detection and summarization approach by disambiguating topic senses.

5 Conclusion

The research described in this paper explores a method for detecting topic words over time in series of documents. The results of extrinsic evaluation showed that integration of LDA and MACD is effective for topic detection.

⁵<http://nlp.stanford.edu/software/CRF-NER.shtml>

⁶<http://catalog.ldc.upenn.edu/LDC2002T31>

References

- J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. 1998. Topic Detection and Tracking Pilot Study Final Report. In *Proc. of the DARPA Broadcast News Transcription and Understanding Workshop*.
- J. Allan, editor. 2003. *Topic Detection and Tracking*. Kluwer Academic Publishers.
- D. M. Blei and J. D. Lafferty. 2006. Dynamic Topic Models. In *Proc. of the 23rd International Conference on Machine Learning*, pages 113–120.
- D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent Dirichlet Allocation. In *The Journal of Machine Learning Research*, volume 3, pages 993–1022.
- A. Celikyilmaz and D. Hakkani-Tur. 2010. A Hybrid Hierarchical Model for Multi-Document Summarization. In *Proc. of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 815–824.
- A. Celikyilmaz and D. Hakkani-Tur. 2011. Discovery of Topically Coherent Sentences for Extractive Summarization. In *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 491–499.
- G. Erkan and D. Radev. 2004. LexPageRank: Prestige in Multi-Document Text Summarization. In *Proc. of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 365–371.
- J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 363–370.
- G. Folino, C. Pizzuti, and G. Spezzano. 2007. An Adaptive Distributed Ensemble Approach to Mine Concept-Drifting Data Streams. In *Proc. of the 19th IEEE International Conference on Tools with Artificial Intelligence*, pages 183–188.
- F. Fukumoto, Y. Suzuki, A. Takasu, and S. Matsuyoshi. 2013. Multi-document summarization based on event and topic detection. In *Proc. of the 6th Language and Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics*, pages 117–121.
- D. He and D. S. Parker. 2010. Topic Dynamics: An Alternative Model of Bursts in Streams of Topics. In *Proc. of the 16th ACM Special Interest Group on Knowledge Discovery and Data Mining*, pages 443–452.
- R. Klinkenberg and T. Joachims. 2000. Detecting Concept Drift with Support Vector Machines. In *Proc. of the 17th International Conference on Machine Learning*, pages 487–494.
- R. Klinkenberg. 2004. Learning Drifting Concepts: Example Selection vs. Example Weighting. *Intelligent Data Analysis*, 8(3):281–300.
- T. Kudo and Y. Matsumoto. 2003. Fast methods for kernel-based text analysis. In *Proc. of 41st Annual Meeting of the Association for Computational Linguistics*, pages 24–31.
- M. M. Lazarescu, S. Venkatesh, and H. H. Bui. 2004. Using Multiple Windows to Track Concept Drift. *Intelligent Data Analysis*, 8(1):29–59.
- W. Li and A. McCallum. 2006. Pachinko Allocation: Dag-Structure Mixture Model of Topic Correlations. In *Proc. of the 23rd International Conference on Machine Learning*, pages 577–584.
- K. Mane and K. Borner. 2004. Mapping Topics and Topic Bursts in PNAS. *Proc. of the National Academy of Sciences of the United States of America*, 101:5287–5290.
- R. Mihalcea and P. Tarau. 2005. Language Independent Extractive Summarization. In *In Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 49–52.
- J. Murphy. 1999. *Technical Analysis of the Financial Markets*. Prentice Hall.
- L. Page, S. Brin, R. Motwani, and T. Winograd. 1998. The Pagerank Citation Ranking: Bringing Order to the Web. In *Technical report, Stanford Digital Libraries*.
- H. Schmid. 1995. Improvements in Part-of-Speech Tagging with an Application to German. In *Proc. of the European chapter of the Association for Computational Linguistics SIGDAT Workshop*.
- M. Scholz. 2007. Boosting Classifiers for Drifting Concepts. *Intelligent Data Analysis*, 11(1):3–28.
- R. Swan and J. Allan. 2000. Automatic Generation of Overview Timelines. In *Proc. of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 38–45.
- K. Toutanova, C. Brockett, M. Gammon, J. Jagarlamudi, H. Suzuki, and L. Vanderwende. 2007. The Phthy Summarization System: Microsoft Research at DUC. In *Proc. of Document Understanding Conference 2007*.
- X. Wan and J. Yang. 2008. Multi-Document Summarization using Cluster-based Link Analysis. In *Proc. of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 299–306.

Content Importance Models for Scoring Writing From Sources

Beata Beigman Klebanov Nitin Madnani Jill Burstein Swapna Somasundaran

Educational Testing Service

660 Rosedale Road

Princeton, NJ 08541

{bbeigmanklebanov, nmadnani, jburstein, ssomasundaran}@ets.org

Abstract

Selection of information from external sources is an important skill assessed in educational measurement. We address an integrative summarization task used in an assessment of English proficiency for non-native speakers applying to higher education institutions in the USA. We evaluate a variety of content importance models that help predict which parts of the source material should be selected by the test-taker in order to succeed on this task.

1 Introduction

Selection and integration of information from external sources is an important academic and life skill, mentioned as a critical competency in the Common Core State Standards for English Language Arts/Literacy: College-ready students will be able to “gather relevant information from multiple print and digital sources, assess the credibility and accuracy of each source, and integrate the information while avoiding plagiarism.”¹

Accordingly, large-scale assessments of writing incorporate tasks that test this skill. One such test requires test-takers to read a passage, then to listen to a lecture discussing the same topic from a different point of view, and to summarize the points made in the lecture, explaining how they cast doubt on points made in the reading. The quality of the information selected from the lecture is emphasized in excerpts from the scoring rubric for this test (below); essays are scored on a 1-5 scale:

Score 5 successfully selects the important information from the lecture and coherently and accurately presents this information in relation to the relevant information presented in the reading.

Score 4 is generally good in selecting the important information from the lecture ..., but it may have a minor omission.

Score 3 contains some important information from the lecture ..., but it may omit one major key point.

Score 2 contains some relevant information from the lecture ... The response significantly omits or misrepresents important points.

Score 1 provides little or no meaningful or relevant coherent content from the lecture.

The ultimate goal of our project is to improve automated scoring of such essays by taking into account the extent to which a response integrates important information from the lecture. This paper reports on the first step aimed at automatically assigning importance scores to parts of the lecture. The next step – developing an essay scoring system using content importance models along with other features of writing quality, will be addressed in future work. A simple essay scoring mechanism will be used for evaluation purposes in this paper, as described in the next section.

2 Design of Experiment

In evaluations of summarization algorithms, it is common practice to derive the gold standard content importance scores from human summaries, as done, for example, in the pyramid method, where the importance of a content element corresponds to the number of reference human summaries that make use of it (Nenkova and Passonneau, 2004). Selection of the appropriate content plays a crucial role in attaining a high score for the essays we consider here, as suggested by the quotes from the scoring rubric in §1, as well as by a corpus study by Plakans and Gebriel (2013). We therefore observe that high-scoring essays can be thought

¹<http://www.corestandards.org/ELA-Literacy/CCRA/W>.

of as high-quality human summaries of the lecture, albeit containing, in addition, references to the reading material and language that contrasts the different viewpoints, making them a somewhat noisy gold standard. On the other hand, since low-scoring essays contain deficient summaries of the lecture, our setup allows for a richer evaluation than typical in studies using gold standard human data only, in that a good model should not only agree with the gold standard human summaries but should also disagree with sub-standard human summaries. We therefore use correlation with essay score to evaluate content importance models.

The evaluation will proceed as follows. Every essay E is responding to a test prompt that contains a lecture L and a reading R . We identify the essay’s overlap with the lecture:

$$O(E, L) = \{x | x \in L, x \in E\} \quad (1)$$

where the exact definition of x , that is, what is taken to be a single unit of information, will be one of the parameters to be studied. The essay is then assigned the following score by the content importance model M :

$$S_M(E) = \frac{\sum_{x \in O(E,L)} w_M(x) \times C(x, E)}{n_E} \quad (2)$$

where $w_M(x)$ is the importance weight assigned by model M to item x in the lecture, $C(x, E)$ is the count of tokens in E that realize the information unit x , and n_E is the number of tokens in the essay. In this paper, the distinction between x and C is that between type and token count of instances of that type.² This simple scoring mechanism quantifies the rate of usage of important information per token in the essay. Finally, we calculate the correlation of scores assigned to essays by model M with scores assigned to the same essays by human graders.

This design ensures that once x is fixed, all the content importance models are evaluated within the same scoring scheme, so any differences in the correlations can be attributed to the differences in the weights assigned by the importance models.

²In the future, we intend to explore more complex realization functions, allowing paraphrase, skip n -grams (as in ROUGE (Lin, 2004)), and other approximate matches, such as misspellings and inflectional variants.

3 Content Importance Models

Our setting can be thought of as a special kind of summarization task. Test-takers are required to summarize the lecture while referencing the reading, making this a hybrid of single- and multi-document summarization, where one source is treated as primary and the other as secondary.

We therefore consider models of content importance that had been found useful in the summarization literature, as well as additional models that utilize a special feature of our scenario: We have hundreds of essays of varying quality responding to any given prompt, as opposed to a typical news summarization scenario where a small number of high quality human summaries are available for a given article. A sample of these essays can be used when developing a content importance model.

We define the following importance models. For all definitions, x is a unit of information in the lecture; $C(x, t)$ is the number of tokens in text t that realize x ; n_L and n_R are the number of tokens in the lecture and the reading, respectively.³

Naïve: $w(x) = 1$. This is a simple overlap model.

Prob: $w(x) = \frac{C(x,L)}{n_L}$, an MLE estimate of the probability that x appears in the lecture. Those x that appear more are more important.

Position: $w(x) = \frac{FP(x)}{n_L}$, where $FP(x)$ is the offset of the first occurrence of x in the lecture. The offset corresponds to the token’s serial number in the text, 1 through n_L .

LectVsRead: $w(x) = \frac{C(x,L)}{n_L} - \frac{C(x,R)}{n_R}$, that is, the difference in the probabilities of occurrence of x in the lecture and in the reading passage that accompanies the lecture. This model attempts to capture the contrastive aspect of importance – the content that is unique to the lecture is more important than the content that is shared by the lecture and the reading.

The following two models capitalize on evidence of use of information in better and worse essays. For estimating these models, we sample, for each prompt, a development set of 750 essays responding to the prompt (that is, addressing a given pair of lecture and reading stimuli). Out of these, we take, for each prompt, all essays at score points

³Prob, Position, and LectVsRead models normalize by n_R and n_L to enable comparison of essays responding to different lecture + reading stimuli (prompts).

4 and 5 (**EGood**) and all essays at score points 1 and 2 (**EBad**). These data do not overlap with the experimental data described in section 4. In both definitions below, e is an essay.

Good: $w(x) = \frac{|\{e \in E_{Good} | x \in e\}|}{|E_{Good}|}$. An x is more important if more good essays use it. Hong and Nenkova (2014) showed that a variant of this measure used on pairs of articles and their abstracts from the New York Times effectively identified words that typically go into summaries, *across topics*. In contrast, our measurements are prompt-specific.

GoodVsBad: $w(x) = \frac{|\{e \in E_{Good} | x \in e\}|}{|E_{Good}|} - \frac{|\{e \in E_{Bad} | x \in e\}|}{|E_{Bad}|}$. An x is more important if good essays use it more than bad essays. To our knowledge, this measure has not been used in the summarization literature, probably because a large sample of human summaries of varying quality is typically not available.

4 Data

We use 116 prompts drawn from an assessment of English proficiency for non-native speakers. Each prompt contains a lecture and a reading passage. For each prompt, we sample about 750 essays. Each essay has an operational score provided by a human grader. Table 1 shows the distribution of essay scores; mean score is 3. Text transcripts of the lectures were used.

Score	1	2	3	4	5
Proportion	0.13	0.18	0.35	0.25	0.09

Table 1: Distribution of essay scores.

5 Results

Independently from the content importance models, we address the effect of the granularity of the unit of information. Intuitively, since all the materials for a given prompt deal with the same topic, we expect large unigram overlaps between lecture and reading, and between good and bad essays, whereas n -grams with larger n can be more distinctive. On the other hand, larger n lead to misses, where an information unit would fail to be identified in an essay due to a paraphrase, thus impairing the ability of the scoring function to use the content importance model effectively.

We therefore evaluate each content importance model for different granularities of the content unit x : n -grams for $n = 1, 2, 3, 4$. Table 2 shows the correlations with essay scores.

Content Importance Model	Pearson’s r			
	n=1	n=2	n=3	n=4
Naïve	<u>0.24</u>	0.27*	0.24	0.20
Prob	0.04	0.14	0.17	0.14
Position	0.22	0.30*	0.26*	0.20
LectVsRead	0.09	0.25*	0.31*	0.26*
Good	0.07	0.15	0.10	0.07
GoodVsBad	0.54*	0.42*	0.32*	0.21

Table 2: Correlations with essay scores attained by content models, for various definitions of information unit (n -grams with $n = 1, 2, 3, 4$). Five top scores are boldfaced. The baseline performance is shown in underlined italics. Correlations that are significantly better ($p < 0.05$) than the naïve $n = 1$ model are marked with an asterisk. We use McNemar (1955, p. 148) test for significance of difference between same-sample correlations. $N = 85, 252$ for all correlations.

6 Discussion

The Naïve model with $n = 1$ can be considered a baseline, corresponding to unweighted word overlap between the lecture and the essay. This model attains a significant positive correlation with essay score ($r = 0.24$), suggesting that, in general, better writers use more material from the lecture.

Our next observation is that the Prob and Good models do not improve over the baseline, that is, their weighting schemes generally assign higher weights to the wrong units. We believe the reason for this is that the most highly used n -grams, in the lecture and in the essays, correspond to general topical and functional elements. The importance of these elements is discounted in the more effective Position, LectVsRead, and GoodVsBad models, highlighting subtler aspects of the lecture.

Next, let us consider the granularity of the units of information. We observe that 4-grams are inferior to trigrams for all models, suggesting that data sparsity is becoming a problem for matching 4-word sequences. For models that assign weight based on one or two sources (lecture, or lecture and reading) – Naïve, Position, LectVsRead – unigram models are generally ineffective, while bi-

gram and trigram models significantly outperform the baseline. We interpret this as suggesting that it is certain particular, detailed aspects of the topical concepts that constitute the important nuggets in the lecture; these are usually realized by multiword sequences.

The GoodVsBad models show a different pattern, obtaining the best performance with a unigram version. These models are sensitive to data sparsity not only when matching essays to the lecture (this problem is common to all models) but also during model building. Recall that the weights in a GoodVsBad model are estimated based on differential use in samples of good and bad essays. The estimation of use-in-a-corpus is more accurate for smaller n , because longer n -grams are more susceptible to paraphrasing, which leads to under-estimation of use. Assuming that paraphrasing behavior of good and bad writers is not the same – in fact, there is corpus evidence that better writers paraphrase more (Burstein et al., 2012) – the resulting inaccuracies might impact the estimation of differential use in a systematic manner, making the $n > 1$ models less effective than the unigrams. Given that (a) the GoodVsBad bigram model is the second best overall in spite of the shortcomings of the estimation process, and (b) that the bigram models worked better than unigram models for all the other content importance models, the GoodVsBad bigram model could probably be improved significantly by using a more flexible information realization mechanism.

To illustrate the information assigned high importance by different models, consider a lecture discussing advantages of fish farming. The top-scoring Good bigrams are topical expressions (*fish farming*), functional bigrams around *fish* and *farming*,⁴ aspects of content dealt with at length in the lecture (*wild fish*, *commercial fishing*), bigrams referencing some of the claims – fish containing *less fat* and being used for *fish meal*. In addition, this model picks out some sequences of function words and punctuation (*of the*, *are not*, *and*, *the*) that suggest that better essays tend to give more detail (hence have more complex noun phrases and coordinated constructions) and to draw contrast.

For the bigram GoodVsBad model, the topical bigram *fish farming* is not in the top 20 bi-

grams. Although some bigrams are shared with the Good model, the GoodVsBad model selects additional details about the claims, such as the contrast between *inedible fish* and *edible fish* that is *eaten by humans*, as well as reference to *chemicals used* in farming and to the claim that wild fish are *already endangered* by other practices.

The most important bigrams according to the LectVsRead model include functional bigrams around *fish* and *farming*, functional sequences (*that the*, *is a*), as well as *commercial fishing* and *edible fish*. Also selected are functional bigrams around *consumption* and *species*, hinting, indirectly, at the edibility differences between species. Finally, this model selects almost all bigrams in *the reading passage makes*, *the reading makes claims that* and *the reading says*. While distinguishing the lecture from the reading, these do not capture topic-relevant content of the lecture.

The GoodVsBad unigram model selects *poultry*, *endangered*, *edible*, *chemicals* among its top 6 unigrams,⁵ effectively touching upon the connection with other farm-raised foods (*poultry*, *chemicals*), with wild fish (*endangered*) and with human benefit (*edible*) that are made in the lecture.

7 Related work

Modern essay scoring systems are complex and cover various aspects of the writing construct, such as grammar, organization, vocabulary (Shermis and Burstein, 2013). The quality of content is often addressed by features that quantify the similarity between the vocabulary used in an essay and reference essays from given score points (Attali and Burstein, 2006; Foltz et al., 2013; Attali, 2011). For example, Attali (2011) proposed a measure of differential use of words in higher and lower scoring essays defined similarly to GoodVsBad, without, however, considering the source text at all. Such features can be thought of as content quality features, as they implicitly assume that writers of better essays use better content. However, there are various kinds of better content, only one of them being selection of important information from the source; other elements of content originate with the writer, such as examples, discourse markers, evaluations, introduction and conclusion, etc. Our approach allows focusing on a particular aspect of content quality, namely, selection of appropriate materials from the source.

⁴such as *that fish*, *of fish*, *farming is*, *fish*

⁵the other two being *fishing* and *used*.

Our results are related to the findings of Gurevich and Deane (2007) who studied the difference between the reading and the lecture in their impact on essay scores for this test. Using data from a single prompt, they showed that the difference between the essay’s average cosine similarity to the reading and its average cosine similarity to the lecture is predictive of the score for non-native speakers of English, thus using a model similar to LectVsRead, although they took all lecture, reading, and essay words into account, in contrast to our model that looks only at n -grams that appear in the lecture. Our study shows that the effectiveness of lecture-reading contrast models for essay scoring generalizes to a large set of prompts. Similarly, Evanini et al. (2013) found that overlap with material that is unique to the lecture (not shared with the reading) was predictive of scores in a spoken source-based question answering task.

In the vast literature on summarization, our work is closest to Hong and Nenkova (2014) who studied models of word importance for multi-document summarization of news. The Prob, Position, and Good models are inspired by their findings of the effectiveness of similar models in their setting. We found that, in our setting, Prob and Good models performed worse than assigning a uniform weight to all words. We note, however, that models from Hong and Nenkova (2014) are not strictly comparable, since their word probability models were calculated after stopword exclusion, and their model that inspired our Good model was defined somewhat differently and validated using content words only. The definition of our Position model and its use in the essay scoring function S (equation 2) correspond to Hong and Nenkova (2014) average first location model for scoring summaries. Differently from their findings, this model is not effective for single words in our setting. Position models over n -grams with $n > 1$ are effective, but their prediction is in the *opposite* direction of that found for the news data – the more important materials tend to appear *later* in the lecture, as indicated by the positive r between average first position and essay score. These findings underscore the importance of paying attention to the genre of the source material when developing summarization systems.

Our summarization task incorporates elements of contrastive opinion summarization (Paul et al., 2010; Kim and Zhai, 2009), since the lecture and

the reading sometimes interpret the same facts in a positive or negative light (for example, the fact that chemicals are used in fish farms is negative if compared to wild fish, but not so if compared to other farm-raised foods like poultry). Relationships between aspect and sentiment (Brody and Elhadad, 2010; Lazaridou et al., 2013) are also relevant, since aspects of the same fact are emphasized with different evaluations (the quantity vs the variety of species that go into fish meal for farmed fish). We hypothesize that units participating in sentiment and aspect contrasts are of higher importance; this is a direction for future work.

8 Conclusion

In this paper, we addressed the task of automatically assigning importance scores to parts of a lecture that is to be summarized as part of an English language proficiency test. We investigated the optimal units of information to which importance should be assigned, as well as a variety of importance scoring models, drawing on the news summarization and essay scoring literature.

We found that bigrams and trigrams were generally more effective than unigrams and 4-grams across importance models, with some exceptions.

We also found that the most effective importance models are those that equate importance of an n -gram with its preferential use in higher-scoring essays than in lower-scoring ones, above and beyond merely looking at the n -grams used in good essays. This demonstrates the utility of using not only gold, high-quality human summaries, but also sub-standard ones when developing content importance models.

Additional importance criteria that are intrinsic to the lecture, as well as those that capture contrast with a different source discussing the same topic, were also found to be reasonably effective. Since different importance models often select different items as most important, we intend to investigate complementarity of the different models.

Finally, our results highlight that the effectiveness of an importance model depends on the genre of the source text. Thus, while a first sentence baseline is very competitive in news summarization, we found that important information tends *not* to be located in the opening sentences in our data (these tend to provide general, introductory information), but appears later on, when more detailed, specific claims are put forward.

References

- Yigal Attali and Jill Burstein. 2006. Automated Essay Scoring With e-rater®V.2. *Journal of Technology, Learning, and Assessment*, 4(3).
- Yigal Attali. 2011. A Differential Word Use Measure for Content Analysis in Automated Essay Scoring. *ETS Research Report*, RR-11-36.
- Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 804–812, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Jill Burstein, Michael Flor, Joel Tetreault, Nitin Madnani, and Steven Holtzman. 2012. Examining Linguistic Characteristics of Paraphrase in Test-Taker Summaries. *ETS Research Report*, RR-12-18.
- Keelan Evanini, Shasha Xie, and Klaus Zechner. 2013. Prompt-based content scoring for automated spoken language assessment. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 157–162, Atlanta, Georgia, June. Association for Computational Linguistics.
- Peter Foltz, Lynn Streeter, Karen Lochbaum, and Thomas Landauer. 2013. Implementation and Application of the Intelligent Essay Assessor. In Mark Shermis and Jill Burstein, editors, *Handbook of automated essay evaluation: Current applications and new directions*, pages 68–88. New York: Routledge.
- Olga Gurevich and Paul Deane. 2007. Document similarity measures to distinguish native vs. non-native essay writers. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 49–52, Rochester, New York, April. Association for Computational Linguistics.
- Kai Hong and Ani Nenkova. 2014. Improving the estimation of word importance for news multi-document summarization. In *The Conference of the European Chapter of the Association for Computational Linguistics*, Gottenberg, Sweden, April. Association for Computational Linguistics.
- Hyun Duk Kim and ChengXiang Zhai. 2009. Generating comparative summaries of contradictory opinions in text. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 385–394, New York, NY, USA. ACM.
- Angeliki Lazaridou, Ivan Titov, and Caroline Sporleder. 2013. A bayesian model for joint unsupervised induction of sentiment, aspect and discourse representations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1630–1639, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Proceedings of ACL workshop: Text summarization branches out*, pages 74–81, Barcelona, Spain, July. Association for Computational Linguistics.
- Quinn McNemar. 1955. *Psychological Statistics*. New York: J. Wiley and Sons, 2nd edition.
- Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Human Language Technologies 2004: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 145–152, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.
- Michael J. Paul, ChengXiang Zhai, and Roxana Girju. 2010. Summarizing contrastive viewpoints in opinionated text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 66–76, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Lia Plakans and Atta Gebril. 2013. Using multiple texts in an integrated writing assessment: Source text use as a predictor of score. *Journal of Second Language Writing*, 22:217–230.
- Mark Shermis and Jill Burstein, editors. 2013. *Handbook of Automated Essay Evaluation: Current Applications and Future Directions*. New York: Routledge.

Chinese Morphological Analysis with Character-level POS Tagging

Mo Shen[†], Hongxiao Liu[‡], Daisuke Kawahara[†], and Sadao Kurohashi[†]

[†]Graduate School of Informatics, Kyoto University, Japan

[‡]School of Computer Science, Fudan University, China

shen@nlp.ist.i.kyoto-u.ac.jp {dk,kuro}@i.kyoto-u.ac.jp
12210240027@fudan.edu.cn

Abstract

The focus of recent studies on Chinese word segmentation, part-of-speech (POS) tagging and parsing has been shifting from words to characters. However, existing methods have not yet fully utilized the potentials of Chinese characters. In this paper, we investigate the usefulness of character-level part-of-speech in the task of Chinese morphological analysis. We propose the first tagset designed for the task of character-level POS tagging. We propose a method that performs character-level POS tagging jointly with word segmentation and word-level POS tagging. Through experiments, we demonstrate that by introducing character-level POS information, the performance of a baseline morphological analyzer can be significantly improved.

1 Introduction

In recent years, the focus of research on Chinese word segmentation, part-of-speech (POS) tagging and parsing has been shifting from words toward characters. Character-based methods have shown superior performance in these tasks compared to traditional word-based methods (Ng and Low, 2004; Nakagawa, 2004; Zhao et al., 2006; Kruengkrai et al., 2009; Xue, 2003; Sun, 2010). Studies investigating the morphological-level and character-level internal structures of words, which treat character as the true atom of morphological and syntactic processing, have demonstrated encouraging results (Li, 2011; Li and Zhou, 2012; Zhang et al., 2013). This line of research has provided great insight in revealing the roles of characters in word formation and syntax of Chinese language.

However, existing methods have not yet fully utilized the potentials of Chinese characters. While Li (2011) pointed out that some characters

Character-level Part-of-Speech	Examples of Verb
verb + noun	投资 (invest : throw + wealth)
noun + verb	心疼 (feel sorry : heart + hurt)
verb + adjective	认清 (realize : recognize + clear)
adjective + verb	痛恨 (hate : pain + hate)
verb + verb	审查 (inspect : examine + review)

Table 1. Character-level POS sequence as a more specified version of word-level POS: an example of verb.

can productively form new words by attaching to existing words, these characters consist only a portion of all Chinese characters and appear in 35% of the words in Chinese Treebank 5.0 (CTB5) (Xue et al., 2005). Zhang (2013) took one step further by investigating the character-level structures of words; however, the machine learning of inferring these internal structures relies on the character forms, which still suffers from data sparseness.

In our view, since each Chinese character is in fact created as a word in origin with complete and independent meaning, it should be treated as the actual minimal morphological unit in Chinese language, and therefore should carry specific part-of-speech. For example, the character “打” (beat) is a verb and the character “破” (broken) is an adjective. A word on the other hand, is either single-character, or a compound formed by single-character words. For example, the verb “打破” (break) can be seen as a compound formed by the two single-character words with the construction “verb + adjective”.

Under this treatment, we observe that words with the same construction in terms of character-level POS tend to also have similar syntactic roles. For example, the words having the con-

struction “verb + adjective” are typically verbs, and those having the construction “adjective + noun” are typically nouns, as shown in the following examples:

- (a) verb : verb + adjective
 “打破”(break) : “打”(beat) + “破”(broken)
 “更新”(update) : “更”(replace) + “新”(new)
 “漂白”(bleach) : “漂”(wash) + “白”(white)
- (b) noun : adjective + noun
 “主题”(theme) : “主”(main) + “题”(topic)
 “新人”(newcomer) : “新”(new) + “人”(person)
 “快车”(express) : “快”(fast) + “车”(car)

This suggests that character-level POS can be used as cues in predicting the part-of-speech of unknown words.

Another advantage of character-level POS is that, the sequence of character-level POS in a word can be seen as a more fine-grained version of word-level POS. An example is shown in Table 1. The five words in this table are very likely to be tagged with the same word-level POS as verb in any available annotated corpora, while it can be commonly agreed among native speakers of Chinese that the syntactic behaviors of these words are different from each other, due to their distinctions in word constructions. For example, verbs having the construction “verb + noun” (e.g. 投资) or “verb + verb” (e.g. 审查) can also be nouns in some context, while others cannot; And verbs having the constructions “verb + adjective” (e.g. 认清) require exact one object argument, while others generally do not. Therefore, compared to word-level POS, the character-level POS can produce information for more expressive features during the learning process of a morphological analyzer.

In this paper, we investigate the usefulness of character-level POS in the task of Chinese morphological analysis. We propose the first tagset designed for the task of character-level POS tagging, based on which we manually annotate the entire CTB5. We propose a method that performs character-level POS tagging jointly with word segmentation and word-level POS tagging. Through experiments, we demonstrate that by introducing character-level POS information, the performance of a baseline morphological analyzer can be significantly improved.

Tag	Part-of-Speech	Example
n	noun	<u>法案</u> /NN (bill)
v	verb	<u>发布</u> /VV (publish)
j	adj./adv.	<u>广阔</u> /VA (vast)
t	numerical	<u>三点一四</u> /CD (3.14)
m	quantifier	<u>一</u> /CD 件/M (a piece of)
d	date	<u>九五年</u> /NT (1995)
k	proper noun	<u>中美</u> /NR (sino-US)
b	prefix	<u>副</u> 市长/NN (vice mayor)
e	suffix	建筑 <u>业</u> /NN (construction industry)
r	transliteration	<u>阿尔帕德</u> /NR (Árpád)
u	punctuation	<u>查尔斯·狄更斯</u> /NR (Charles Dickens)
f	foreign chars	<u>X</u> 射线/NN (X-ray)
o	onomatopoeia	<u>隆隆</u> /AD (rumble)
s	surname	<u>王</u> 新民/NR (Wang Xinmin)
p	pronoun	<u>他们</u> /PN (they)
c	other functional	<u>用于</u> /VV (be used for)

Table 2. Tagset for character-level part-of-speech tagging. The underlined characters in the examples correspond to the tags on the left-most column. The CTB-style word-level POS are also shown for the examples.

2 Character-level POS Tagset

We propose a tagset for the task of character-level POS tagging. This tagset contains 16 tags, as illustrated in Table 2. The tagset is designed by treating each Chinese character as a single-character word, and each (multi-character) word as a phrase of single-character words. Some of these tags are directly derived from the commonly accepted word-level part-of-speech, such as noun, verb, adjective and adverb. It should be noted that, for single-character words, the difference between adjective and adverb can almost be ignored, because for any of such words that can be used as an adjective, it usually can also be used as an adverb. Therefore, we have merged these two tags into one.

On the other hand, some other tags are designed specifically for characters, such as transliteration, surname, prefix and suffix. Unlike some Asian languages such as Japanese, there is no explicit character set in Chinese that are used exclusively for expressing names of foreign persons, places or organizations. However, some characters are used much more frequently than others in these situations. For example, in the person’s name “阿尔帕德” (Árpád), all the four characters can be frequently observed in words

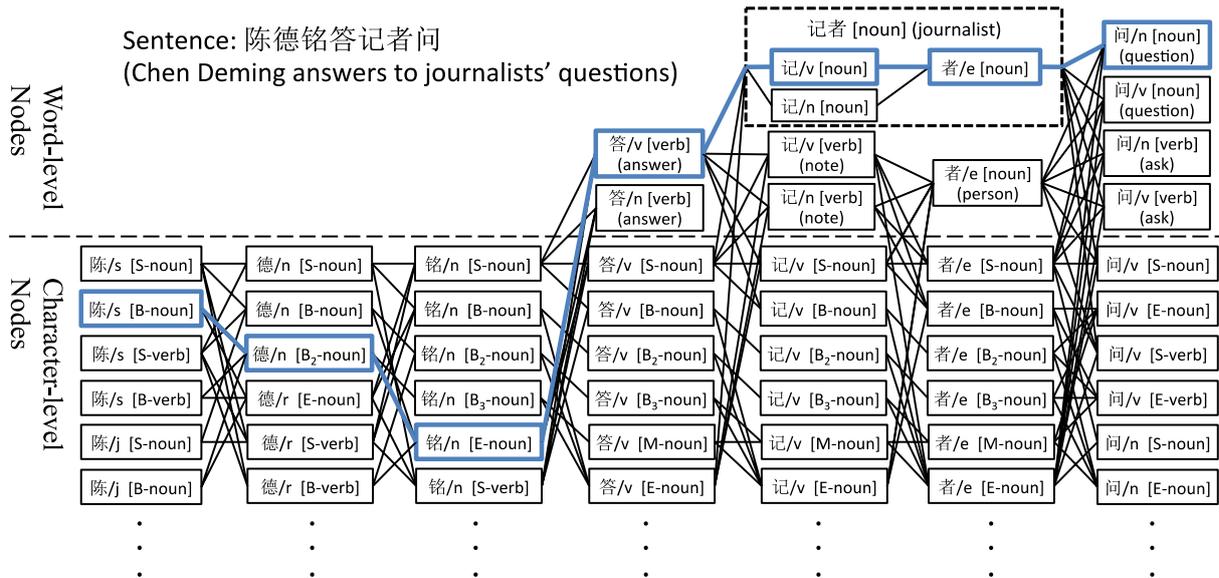


Figure 1. A Word-character hybrid lattice of a Chinese sentence. Correct path is represented by blue bold lines.

Word Length	1	2	3	4	5	6	7 or more
Tags	<i>S</i>	<i>BE</i>	<i>BB₂E</i>	<i>BB₂B₃E</i>	<i>BB₂B₃ME</i>	<i>BB₂B₃MME</i>	<i>BB₂B₃M...ME</i>

Table 3. Word representation with a 6-tag tagset: *S, B, B₂, B₃, M, E*

of transliterations. Similarly, surnames in Chinese are also drawn from a set of limited number of characters. We therefore assign specific tags for this kind of character sets. The tags for prefixes and suffixes are motivated by the previous studies (Li, 2011; Li and Zhou, 2012).

We have annotated character-level POS for all words in CTB5¹. Fortunately, character-level POS in most words are independent of context, which means it is sufficient to annotate word forms unless there is an ambiguity. The annotation was conducted by two persons, where each one of them was responsible for about 70% of the documents in the corpus. The redundancy was set for the purposes of style unification and quality control, on which we find that the inter-annotator agreement is 96.2%. Although the annotation also includes the test set, we blind this portion in all the experiments.

3 Chinese Morphological Analysis with Character-level POS

3.1 System Description

Previous studies have shown that jointly processing word segmentation and POS tagging is preferable to pipeline processing, which can propagate errors (Nakagawa and Uchimoto, 2007; Kruengkrai et al., 2009). Based on these studies, we propose a word-character hybrid model which can also utilize the character-level POS information. This hybrid model constructs a lattice that consists of word-level and character-level nodes from a given input sentence. Word-level nodes correspond to words found in the system's lexicon, which has been compiled from training data. Character-level nodes have special tags called position-of-character (POC) that indicate the word-internal position (Asahara, 2003; Nakagawa, 2004). We have adopted the 6-tag tagset, which (Zhao et al., 2006) reported to be optimal. This tagset is illustrated in Table 3.

Figure 2 shows an example of a lattice for the Chinese sentence: “陈德铭答记者问” (Chen Deming answers to journalists' questions). The correct path is marked with blue bold lines. The

¹ <http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?CharPosCN>

Category	Template	Condition
Baseline-unigram	$\langle w_0, p_0 \rangle \langle w_0, p_0 \rangle \langle l_0, p_0 \rangle \langle \text{begin}(w_0), p_0 \rangle \langle \text{end}(w_0), p_0 \rangle$	W_0
	$\langle \text{begin}(w_0), \text{end}(w_0), p_0 \rangle$	
	$\langle c_{-2}, p_0 \rangle \langle c_{-1}, p_0 \rangle \langle c_0, p_0 \rangle \langle c_1, p_0 \rangle \langle c_2, p_0 \rangle$	C_0
	$\langle c_{-2}, c_{-1}, p_0 \rangle \langle c_{-1}, c_0, p_0 \rangle \langle c_0, c_1, p_0 \rangle \langle c_1, c_2, p_0 \rangle \langle c_{-1}, c_1, p_0 \rangle$	
Baseline-bigram	$\langle w_{-1}, w_0 \rangle \langle p_{-1}, p_0 \rangle \langle w_{-1}, p_0 \rangle \langle p_{-1}, w_0 \rangle \langle w_{-1}, p_{-1}, w_0 \rangle \langle w_{-1}, w_0, p_0 \rangle$	$W_{-1} \times W_0$
	$\langle w_{-1}, p_{-1}, p_0 \rangle \langle p_{-1}, w_0, p_0 \rangle \langle w_{-1}, p_{-1}, w_0, p_0 \rangle \langle l_{-1}, p_{-1}, l_0 \rangle \langle l_{-1}, l_0, p_0 \rangle$	
	$\langle l_{-1}, p_{-1}, p_0 \rangle \langle p_{-1}, l_0, p_0 \rangle \langle l_{-1}, p_{-1}, l_0, p_0 \rangle \langle \text{end}(w_{-1}), p_0 \rangle$	
	$\langle p_{-1}, \text{begin}(w_0) \rangle \langle \text{end}(w_{-1}), p_{-1}, p_0 \rangle \langle p_{-1}, \text{begin}(w_0), p_0 \rangle$	
	$\langle c_{-1}, c_0 \rangle \langle p_{-1}, p_0 \rangle \langle c_{-1}, p_{-1}, c_0 \rangle \langle c_{-1}, c_0, p_0 \rangle$	$C_{-1} \times C_0$
	$\langle c_{-1}, p_{-1}, p_0 \rangle \langle p_{-1}, c_0, p_0 \rangle \langle c_{-1}, p_{-1}, c_0, p_0 \rangle$	
	$\langle p_{-1}, p_0 \rangle$	Otherwise
Proposed-unigram	$\langle \text{CP}(c_0), p_0 \rangle$	C_0
Proposed-bigram	$\langle \text{CP}_{\text{pair}}(w_{-1}), p_0 \rangle \langle \text{CP}_{\text{pair}}(w_{-1}), p_{-1}, p_0 \rangle$	$W_{-1} \times N_0$
	$\langle \text{CP}_{\text{all}}(w_{-1}), p_0 \rangle \langle \text{CP}_{\text{all}}(w_{-1}), p_{-1}, p_0 \rangle$	
	$\langle \text{CP}_{\text{pair}}(w_{-1}), p_{-1}, \text{CP}(c_0) \rangle \langle \text{CP}_{\text{pair}}(w_{-1}), \text{CP}(c_0), p_0 \rangle$	$W_{-1} \times C_0$
	$\langle \text{CP}_{\text{all}}(w_{-1}), p_{-1}, \text{CP}(c_0) \rangle \langle \text{CP}_{\text{all}}(w_{-1}), \text{CP}(c_0), p_0 \rangle$	
	$\langle p_{-1}, \text{CP}(c_0) \rangle \langle p_{-1}, \text{CP}(c_0), p_0 \rangle$	$N_{-1} \times C_0$
	$\langle \text{CP}(c_{-1}), p_0 \rangle \langle \text{CP}(c_{-1}), p_{-1}, p_0 \rangle$	$C_{-1} \times N_0$
	$\langle \text{CP}(c_{-1}), p_{-1}, \text{CP}(c_0) \rangle \langle \text{CP}(c_{-1}), \text{CP}(c_0), p_0 \rangle \langle \text{CP}(c_{-1}), p_{-1}, \text{CP}(c_0), p_0 \rangle$	$C_{-1} \times C_0$

Table 4. Feature templates. The ‘‘Condition’’ column describes when to apply the templates: W_{-1} and W_0 denote the previous and the current word-level node; C_{-1} and C_0 denote the previous and the current character-level node; N_{-1} and N_0 denote the previous and the current node of any types. Word-level nodes represent known words that can be found in the system’s lexicon.

upper part of the lattice (word-level nodes) represents known words, where each node carries information such as character form, character-level POS, and word-level POS. A word that contains multiple characters is represented by a sub-lattice (the dashed rectangle in the figure), where a path stands for a possible sequence of character-level POS for this word. For example, the word ‘‘记者’’ (journalist) has two possible paths of character-level POS: ‘‘verb + suffix’’ and ‘‘noun + suffix’’. Nodes that are inside a sub-lattice cannot be linked to nodes that are outside, except from the boundaries. The lower part of the lattice (character-level nodes) represents unknown words, where each node carries a position-of-character tag, in addition to other types of information that can also be found on a word-level node. A sequence of character-level nodes are considered as an unknown word if and only if the sequence of POC tags forms one of the cases listed in Table 3. This table also illustrates the permitted transitions between adjacent character-level nodes. We use the standard dynamic programming technique to search for the best path in the lattice. We use the averaged perceptron (Collins, 2002), an efficient online learning algorithm, to train the model.

3.2 Features

We show the feature templates of our model in Table 4. The features consist of two categories:

baseline features, which are modified from the templates proposed in (Kruengkrai et al., 2009); and proposed features, which encode character-level POS information.

Baseline features: For word-level nodes that represent known words, we use the symbols w , p and l to denote the word form, POS tag and length of the word, respectively. The functions $\text{begin}(w)$ and $\text{end}(w)$ return the first and last character of w . If w has only one character, we omit the templates that contain $\text{begin}(w)$ or $\text{end}(w)$. We use the subscript indices 0 and -1 to indicate the current node and the previous node during a Viterbi search, respectively. For character-level nodes, c denotes the surface character, and p denotes the combination of POS and POC (position-of-character) tags.

Proposed features: For word-level nodes, the function $\text{CP}_{\text{pair}}(w)$ returns the pair of the character-level POS tags of the first and last characters of w , and $\text{CP}_{\text{all}}(w)$ returns the sequence of character-level POS tags of w . If either the pair or the sequence of character-level POS is ambiguous, which means there are multiple paths in the sub-lattice of the word-level node, then the values on the current best path (with local context) during the Viterbi search will be returned. If w has only one character, we omit the templates that contain $\text{CP}_{\text{pair}}(w)$. For character-level nodes, the function $\text{CP}(c)$ returns its character-level POS. The subscript indices 0 and -1 as well as

other symbols stand for the same meaning as they are in the baseline features.

4 Evaluation

4.1 Settings

To evaluate our proposed method, we have conducted two sets of experiments on CTB5: word segmentation, and joint word segmentation and word-level POS tagging. We have adopted the same data division as in (Jiang et al., 2008a; Jiang et al., 2008b; Kruengkrai et al., 2009; Zhang and Clark, 2010; Sun, 2011): the training set, dev set and test set have 18,089, 350 and 348 sentences, respectively. The models applied on all test sets are those that result in the best performance on the CTB5 dev set.

We have annotated character-level POS information for all 508,768 word tokens in CTB5. As mentioned in section 2, we blind the annotation in the test set in all the experiments. To learn the characteristics of unknown words, we built the system’s lexicon using only the words in the training data that appear at least 3 times. We applied a similar strategy in building the lexicon for character-level POS, where the threshold we choose is 2. These thresholds were tuned using the development data.

We have used precision, recall and the F-score to measure the performance of the systems. Precision (P) is defined as the percentage of output tokens that are consistent with the gold standard test data, and recall (R) is the percentage of tokens in the gold standard test data that are recognized in the output. The balanced F-score (F) is defined as $\frac{2 \cdot P \cdot R}{P + R}$.

4.2 Experimental Results

We compare the performance between a baseline model and our proposed approach. The results of the word segmentation experiment and the joint experiment of segmentation and POS tagging are shown in Table 5(a) and Table 5(b), respectively. Each row in these tables shows the performance of the corresponding system. “CharPos” stands for our proposed model which has been described in section 3. “Baseline” stands for the same model except it only enables features from the baseline templates.

The results show that, while the differences between the baseline model and the proposed model in word segmentation accuracies are small, the proposed model achieves significant improvement in the experiment of joint segmentati-

(a) Word Segmentation Results			
System	P	R	F
Baseline	97.48	98.44	97.96
CharPOS	97.55	98.51	98.03

(b) Joint Segmentation and POS Tagging Results			
System	P	R	F
Baseline	93.01	93.95	93.48
CharPOS	93.42	94.18	93.80

Table 5. Experimental results on CTB5.

System	Segmentation	Joint
Baseline	97.96	93.48
CharPOS	98.03	93.80
Jiang2008a	97.85	93.41
Jiang2008b	97.74	93.37
Kruengkrai2009	97.87	93.67
Zhang2010	97.78	93.67
Sun2011	98.17	94.02

Table 6. Comparison with previous studies on CTB5.

on and POS tagging². This suggests that our proposed method is particularly effective in predicting the word-level POS, which is consistent with our observations mentioned in section 1.

In Table 6 we compare our approach with morphological analyzers in previous studies. The accuracies of the systems in previous work are directly taken from the original paper. As the results show, despite the fact that the performance of our baseline model is relatively weak in the joint segmentation and POS tagging task, our proposed model achieves the second-best performance in both segmentation and joint tasks.

5 Conclusion

We believe that by treating characters as the true atoms of Chinese morphological and syntactic analysis, it is possible to address the out-of-vocabulary problem that word-based methods have been long suffered from. In our error analysis, we believe that by exploring the character-level POS and the internal word structure (Zhang et al., 2013) at the same time, it is possible to further improve the performance of morphological analysis and parsing. We will address these issues in our future work.

² $p < 0.05$ in McNemar’s test.

Reference

- Masayuki Asahara. 2003. Corpus-based Japanese Morphological Analysis. Nara Institute of Science and Technology, Doctor's Thesis.
- Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In Proceedings of EMNLP, pages 1–8.
- Wenbin Jiang, Liang Huang, Qun Liu, and Yajuan Lü. 2008a. A Cascaded Linear Model for Joint Chinese Word Segmentation and Part-of-speech Tagging. In Proceedings of ACL.
- Wenbin Jiang, Haitao Mi, and Qun Liu. 2008b. Word Lattice Reranking for Chinese Word Segmentation and Part-of-speech Tagging. In Proceedings of COLING.
- Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiyou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An Error-Driven Word-Character Hybrid Model for Joint Chinese Word Segmentation and POS Tagging. In Proceedings of ACL-IJCNLP, pages 513-521.
- Zhongguo Li. 2011. Parsing the Internal Structure of Words: A New Paradigm for Chinese Word Segmentation. In Proceedings of ACL-HLT, pages 1405–1414.
- Zhongguo Li and Guodong Zhou. 2012. Unified Dependency Parsing of Chinese Morphological and Syntactic Structures. In Proceedings of EMNLP, pages 1445–1454.
- Hwee Tou Ng and Jin Kiat Low. 2004. Chinese Part-of-speech Tagging: One-at-a-time or All-at-once? Word-based or Character-based? In Proceedings of EMNLP, pages 277–284.
- Tetsuji Nakagawa. 2004. Chinese and Japanese word segmentation using word-level and character-level information. In Proceedings of COLING, pages 466–472.
- Tetsuji Nakagawa and Kiyotaka Uchimoto. 2007. Hybrid Approach to Word Segmentation and Pos Tagging. In Proceedings of ACL Demo and Poster Sessions, pages 217-220.
- Weiwei Sun. 2010. Word-based and Character-based Word Segmentation Models: Comparison and Combination. In Proceedings of COLING Poster Sessions, pages 1211–1219.
- Weiwei Sun. 2011. A Stacked Sub-word Model for Joint Chinese Word Segmentation and Part-of-speech Tagging. In Proceedings of ACL-HLT, pages 1385–1394.
- Nianwen Xue. 2003. Chinese Word Segmentation as Character Tagging. In International Journal of Computational Linguistics and Chinese Language Processing.
- Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The Penn Chinese Treebank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering*, 11(2):207–238.
- Hai Zhao, Chang-Ning Huang, Mu Li, and Bao-Liang Lu. 2006. Effective Tag Set Selection in Chinese Word Segmentation via Conditional Random Field Modeling. In Proceedings of PACLIC, pages 87-94.
- Meishan Zhang, Yue Zhang, Wanxiang Che, and Ting Liu. 2013. Chinese Parsing Exploiting Characters. In Proceedings of ACL, page 125-134.
- Yue Zhang and Stephen Clark. 2010. A Fast Decoder for Joint Word Segmentation and POS-tagging Using a Single Discriminative Model. In Proceedings of EMNLP, pages 843–852.

Part-of-Speech Tagging using Conditional Random Fields: Exploiting Sub-Label Dependencies for Improved Accuracy

Miikka Silfverberg^a Teemu Ruokolainen^b Krister Lindén^a Mikko Kurimo^b

^a Department of Modern Languages, University of Helsinki,
firstname.lastname@helsinki.fi

^b Department of Signal Processing and Acoustics, Aalto University,
firstname.lastname@aalto.fi

Abstract

We discuss part-of-speech (POS) tagging in presence of large, fine-grained label sets using conditional random fields (CRFs). We propose improving tagging accuracy by utilizing dependencies within sub-components of the fine-grained labels. These sub-label dependencies are incorporated into the CRF model via a (relatively) straightforward feature extraction scheme. Experiments on five languages show that the approach can yield significant improvement in tagging accuracy in case the labels have sufficiently rich inner structure.

1 Introduction

We discuss part-of-speech (POS) tagging using the well-known conditional random field (CRF) model introduced originally by Lafferty et al. (2001). Our focus is on scenarios, in which the POS labels have a *rich inner structure*. For example, consider

PRON+1SG	V+NON3SG+PRES	N+SG	
I	like	ham	,

where the *compound* labels PRON+1SG, V+NON3SG+PRES, and N+SG stand for pronoun first person singular, verb non-third singular present tense, and noun singular, respectively. Fine-grained labels occur frequently in morphologically complex languages (Erjavec, 2010; Haverinen et al., 2013).

We propose improving tagging accuracy by utilizing dependencies within the *sub-labels* (PRON, 1SG, V, NON3SG, N, and SG in the above example) of the compound labels. From a technical perspective, we accomplish this by making use of the fundamental ability of the CRFs to incorporate arbitrarily defined feature functions. The newly-defined features are expected to alleviate data spar-

city problems caused by the fine-grained labels. Despite the (relative) simplicity of the approach, we are unaware of previous work exploiting the sub-labels to the extent presented here.

We present experiments on five languages (English, Finnish, Czech, Estonian, and Romanian) with varying POS annotation granularity. By utilizing the sub-labels, we gain significant improvement in model accuracy given a sufficiently fine-grained label set. Moreover, our results indicate that exploiting the sub-labels can yield larger improvements in tagging compared to increasing model order.

The rest of the paper is organized as follows. Section 2 describes the methodology. Experimental setup and results are presented in Section 3. Section 4 discusses related work. Lastly, we provide conclusions on the work in Section 5.

2 Methods

2.1 Conditional Random Fields

The (unnormalized) CRF model (Lafferty et al., 2001) for a sentence $x = (x_1, \dots, x_{|x|})$ and a POS sequence $y = (y_1, \dots, y_{|x|})$ is defined as

$$p(y|x; \mathbf{w}) \propto \prod_{i=n}^{|x|} \exp(\mathbf{w} \cdot \phi(y_{i-n}, \dots, y_i, x, i)), \quad (1)$$

where n denotes the model order, \mathbf{w} the model parameter vector, and ϕ the feature extraction function. We denote the tag set as \mathcal{Y} , that is, $y_i \in \mathcal{Y}$ for $i \in 1 \dots |x|$.

2.2 Baseline Feature Set

We first describe our baseline feature set $\{\phi_j(y_{i-1}, y_i, x, i)\}_{j=1}^{|\phi|}$ by defining *emission* and *transition* features. The emission feature set associates properties of the sentence position i with

the corresponding label as

$$\{\chi_j(x, i) \mathbb{1}(y_i = y'_i) \mid j \in 1 \dots |\mathcal{X}|, \forall y'_i \in \mathcal{Y}\}, \quad (2)$$

where the function $\mathbb{1}(q)$ returns one if and only if the proposition q is true and zero otherwise, that is

$$\mathbb{1}(y_i = y'_i) = \begin{cases} 1 & \text{if } y_i = y'_i \\ 0 & \text{otherwise} \end{cases}, \quad (3)$$

and $\mathcal{X} = \{\chi_j(x, i)\}_{j=1}^{|\mathcal{X}|}$ is the set of functions characterizing the word position i . Following the classic work of Ratnaparkhi (1996), our \mathcal{X} comprises simple binary functions:

1. Bias (always active irrespective of input).
2. Word forms x_{i-2}, \dots, x_{i+2} .
3. Prefixes and suffixes of the word form x_i up to length $\delta_{suf} = 4$.
4. If the word form x_i contains (one or more) capital letter, hyphen, dash, or digit.

Binary functions have a return value of either zero (inactive) or one (active). Meanwhile, the transition features

$$\{\mathbb{1}(y_{i-k} = y'_{i-k}) \dots \mathbb{1}(y_i = y'_i) \mid y'_{i-k}, \dots, y'_i \in \mathcal{Y}, \forall k \in 1 \dots n\} \quad (4)$$

capture dependencies between adjacent labels irrespective of the input x .

2.2.1 Expanded Feature Set Leveraging Sub-Label Dependencies

The baseline feature set described above can yield a high tagging accuracy given a conveniently simple label set, exemplified by the tagging results of Collins (2002) on the Penn Treebank (Marcus et al., 1993). (Note that conditional random fields correspond to discriminatively trained hidden Markov models and Collins (2002) employs the latter terminology.) However, it does to some extent overlook some beneficial dependency information in case the labels have a rich sub-structure. In what follows, we describe expanded feature sets which explicitly model the sub-label dependencies.

We begin by defining a function $\mathcal{P}(y_i)$ which partitions any label y_i into its sub-label components and returns them in an unordered set. For example, we could define $\mathcal{P}(\text{PRON}+1+\text{SG}) =$

$\{\text{PRON}, 1, \text{SG}\}$. (Label partitions employed in the experiments are described in Section 3.2.) We denote the set of all sub-label components as \mathcal{S} .

Subsequently, instead of defining only (2), we additionally associate the feature functions \mathcal{X} with all sub-labels $s \in \mathcal{S}$ by defining

$$\{\chi_j(x, i) \mathbb{1}(s \in \mathcal{P}(y_i)) \mid \forall j \in 1 \dots |\mathcal{X}|, \forall s \in \mathcal{S}\}, \quad (5)$$

where $\mathbb{1}(s \in \mathcal{P}(y_i))$ returns one in case s is in $\mathcal{P}(y_i)$ and zero otherwise. Second, we exploit *sub-label transitions* using features

$$\{\mathbb{1}(s_{i-k} \in \mathcal{P}(y_{i-k})) \dots \mathbb{1}(s_i \in \mathcal{P}(y_i)) \mid \forall s_{i-k}, \dots, s_i \in \mathcal{S}, \forall k \in 1 \dots m\}. \quad (6)$$

Note that we define the sub-label transitions up to order m , $1 \leq m \leq n$, that is, an n th-order CRF model is not obliged to utilize sub-label transitions all the way up to order n . This is because employing high-order sub-label transitions may potentially cause overfitting to training data due to substantially increased number of features (equivalent to the number of model parameters, $|\mathbf{w}| = |\phi|$). For example, in a second-order ($n = 2$) model, it might be beneficial to employ the sub-label emission feature set (5) and first-order sub-label transitions while discarding second-order sub-label transitions. (See the experimental results presented in Section 3.)

In the remainder of this paper, we use the following notations.

1. A standard CRF model incorporating (2) and (4) is denoted as $\text{CRF}(n,-)$.
2. A CRF model incorporating (2), (4), and (5) is denoted as $\text{CRF}(n,0)$.
3. A CRF model incorporating (2), (4), (5), and (6) is denoted as $\text{CRF}(n,m)$.

2.3 On Linguistic Intuition

This section aims to provide some intuition on the types of linguistic phenomena that can be captured by the expanded feature set. To this end, we consider an example on the plural number in Finnish.

First, consider the plural nominative word form *kissat* (*cats*) where the plural number is denoted by the 1-suffix *-t*. Then, by employing the features (2), the suffix *-t* is associated solely with the compound label $\text{NOMINATIVE}+\text{PLURAL}$. However, by incorporating the expanded feature set (5), *-t*

will also be associated to the sub-label PLURAL. This can be useful because, in Finnish, also adjectives and numerals are inflected according to number and denote the plural number with the suffix *-t* (Hakulinen et al., 2004, §79). Therefore, one can exploit *-t* to predict the plural number also in words such as *mustat* (plural of *black*) with a compound analysis ADJECTIVE+PLURAL.

Second, consider the number agreement (congruence). For example, in the sentence fragment *mustat kissat juoksevat* (*black cats are running*), the words *mustat* and *kissat* share the plural number. In other words, the analyses of both *mustat* and *kissat* are required to contain the sub-label PLURAL. This short-span dependency between sub-labels will be captured by a first-order sub-label transition feature included in (6).

Lastly, we note that the feature expansion sets (5) and (6) will, naturally, capture any short-span dependencies within the sub-labels irrespective if the dependencies have a clear linguistic interpretation or not.

3 Experiments

3.1 Data

For a quick overview of the data sets, see Table 1.

Penn Treebank. The English Penn Treebank (Marcus et al., 1993) is divided into 25 sections of newswire text extracted from the Wall Street Journal. We split the data into training, development, and test sets using the sections 0-18, 19-21, and 22-24, according to the standardly applied division introduced by Collins (2002).

Turku Dependency Treebank. The Finnish Turku Dependency Treebank (Haverinen et al., 2013) contains text from 10 different domains. The treebank does not have default partition to training and test sets. Therefore, from each 10 consecutive sentences, we assign the 9th and 10th to the development set and the test set, respectively. The remaining sentences are assigned to the training set.

Multext-East. The third data we consider is the multilingual Multext-East (Erjavec, 2010) corpus, from which we utilize the Czech, Estonian and Romanian sections. The corpus corresponds to translations of the novel *1984* by George Orwell. We apply the same data splits as for Turku Dependency Treebank.

lang.	train.	dev.	test	tags	train. tags
Eng	38,219	5,527	5,462	45	45
Rom	5,216	652	652	405	391
Est	5,183	648	647	413	408
Cze	5,402	675	675	955	908
Fin	5,043	630	630	2,355	2,141

Table 1: Overview on data. The training (train.), development (dev.) and test set sizes are given in sentences. The columns titled *tags* and *train. tags* correspond to total number of tags in the data set and number of tags in the training set, respectively.

3.2 Label Partitions

This section describes the employed compound label splits. The label splits for all data sets are submitted as data file attachments. All the splits are performed *a priori* to model learning, that is, we do not try to optimize them on the development sets.

The POS labels in the Penn Treebank are split in a way which captures relevant inflectional categories, such as tense and number. Consider, for example, the split for the present tense third singular verb label $\mathcal{P}(\text{VBZ}) = \{\text{VB}, \text{Z}\}$.

In the Turku Dependency Treebank, each morphological tag consists of sub-labels marking word-class, relevant inflectional categories, and their respective values. Each inflectional category, such as case or tense, combined with its value, such as nominative or present, constitutes one sub-label. Consider, for example, the split for the singular, adessive noun $\mathcal{P}(\text{N+CASE_ADE+NUM_SG}) = \{\text{POS}_\text{N}, \text{CASE}_\text{ADE}, \text{NUM}_\text{SG}\}$.

The labeling scheme employed in the Multext-East data set represents a considerably different annotation approach compared to the Penn and Turku Treebanks. Each morphological analysis is a sequence of feature markers, for example Pw3-r. The first feature marker (P) denotes word class and the rest (w, 3, and r) encode values of inflectional categories relevant for that word class. A feature marker may correspond to several different values depending on word class and its position in the analysis. Therefore it becomes rather difficult to split the labels into similar pairs of inflectional category and value as we are able to do for the Turku Dependency Treebank. Since the interpretation of a feature marker depends on its position in the analysis and the word class, the markers have to be numbered and appended with the

word class marker. For example, consider the split $\mathcal{P}(\text{Pw3-r}) = \{0 : \text{P}, 1 : \text{Pw}, 2 : \text{P3}, 5 : \text{Pr}\}$.

3.3 CRF Model Specification

We perform experiments using first-order and second-order CRFs with zeroth-order and first-order sub-label features. Using the notation introduced in Section 2, the employed models are CRF(1,-), CRF(1,1), CRF(2,-), CRF(2,0), and CRF(2,1). We do not report results using CRF(2,2) since, based on preliminary experiments, this model overfits on all languages.

The CRF model parameters are estimated using the averaged perceptron algorithm (Collins, 2002). The model parameters are initialized with a zero vector. We evaluate the latest averaged parameters on the held-out development set after each pass over the training data and terminate training if no improvement in accuracy is obtained during three last passes. The best-performing parameters are then applied on the test instances.

We accelerate the perceptron learning using beam search (Zhang and Clark, 2011). The beam width, b , is optimized separately for each language on the development sets by considering $b = 1, 2, 4, 8, 16, 32, 64, 128$ until the model accuracy does not improve by at least 0.01 (absolute).

Development and test instances are decoded using Viterbi search in combination with the tag dictionary approach of Ratnaparkhi (1996). In this approach, candidate tags for known word forms are limited to those observed in the training data. Meanwhile, word forms that were unseen during training consider the full label set.

3.4 Software and Hardware

The experiments are run on a standard desktop computer (Intel Xeon E5450 with 3.00 GHz and 64 GB of memory). The methods discussed in Section 2 are implemented in C++.

3.5 Results

The obtained tagging accuracies and training times are presented in Table 2. The times include running the averaged perceptron algorithm and evaluation of the development sets. The column labeled *it.* corresponds to the number of passes over the training data made by the perceptron algorithm before termination. We summarize the results as follows.

First, compared to standard feature extraction approach, employing the sub-label transition fea-

tures resulted in improved accuracy on all languages apart from English. The differences were statistically significant on Czech, Estonian, and Finnish. (We establish statistical significance (with confidence level 0.95) using the standard 1-sided Wilcoxon signed-rank test performed on 10 randomly divided, non-overlapping subsets of the complete test sets.) This results supports the intuition that the sub-label features should be most useful in presence of large, fine-grained label sets, in which case the learning is most affected by data sparsity.

Second, on all languages apart from English, employing a first-order model with sub-label features yielded higher accuracy compared to a second-order model with standard features. The differences were again statistically significant on Czech, Estonian, and Finnish. This result suggests that, compared to increasing model order, exploiting the sub-label dependencies can be a preferable approach to improve the tagging accuracy.

Third, applying the expanded feature set inevitably causes some increase in the computational cost of model estimation. However, as shown by the running times, this increase is not prohibitive.

4 Related Work

In this section, we compare the approach presented in Section 2 to two prior systems which attempt to utilize sub-label dependencies in a similar manner.

Smith et al. (2005) use a CRF-based system for tagging Czech, in which they utilize expanded emission features similar to our (5). However, they do not utilize the full expanded transition features (6). More specifically, instead of utilizing a single chain as in our approach, Smith et al. employ five parallel structured chains. One of the chains models the sequence of word-class labels such as noun and adjective. The other four chains model gender, number, case, and lemma sequences, respectively. Therefore, in contrast to our approach, their system does not capture cross-dependencies between inflectional categories, such as the dependence between the word-class and case of adjacent words. Unsurprisingly, Smith et al. fail to achieve improvement over a generative HMM-based POS tagger of Hajič (2001). Meanwhile, our system outperforms the generative trigram tagger HunPos (Halácsy et al., 2007) which is an im-

model	it.	time (min)	acc.	OOV.
<i>English</i>				
CRF(1, -)	8	9	97.04	88.65
CRF(1, 0)	6	17	97.02	88.44
CRF(1, 1)	8	22	97.02	88.82
CRF(2, -)	9	15	97.18	88.82
CRF(2, 0)	11	36	97.17	89.23
CRF(2, 1)	8	27	97.15	89.04
<i>Romanian</i>				
CRF(1, -)	14	29	97.03	85.01
CRF(1, 0)	13	68	96.96	84.59
CRF(1, 1)	16	146	97.24	85.94
CRF(2, -)	7	19	97.08	85.21
CRF(2, 0)	18	99	97.02	85.42
CRF(2, 1)	12	118	97.29	86.25
<i>Estonian</i>				
CRF(1, -)	15	28	93.39	78.66
CRF(1, 0)	17	66	93.81	80.44
CRF(1, 1)	13	129	93.77	79.37
CRF(2, -)	15	30	93.48	77.13
CRF(2, 0)	13	53	93.78	79.60
CRF(2, 1)	16	105	94.01	79.53
<i>Czech</i>				
CRF(1, -)	6	28	89.28	70.90
CRF(1, 0)	10	112	89.94	74.44
CRF(1, 1)	10	365	90.78	76.83
CRF(2, -)	19	91	89.81	72.44
CRF(2, 0)	13	203	90.35	76.37
CRF(2, 1)	24	936	91.00	77.75
<i>Finnish</i>				
CRF(1, -)	10	80	87.37	59.29
CRF(1, 0)	13	249	88.58	63.46
CRF(1, 1)	12	474	88.41	62.63
CRF(2, -)	11	106	86.74	56.96
CRF(2, 0)	13	272	88.52	63.46
CRF(2, 1)	12	331	88.68	63.62

Table 2: Results.

proved open-source implementation of the well-known TnT tagger of Brants (2000). The obtained HunPos results are presented in Table 3.

	Eng	Rom	Est	Cze	Fin
HunPos	96.58	96.96	92.76	89.57	85.77

Table 3: Results using a generative HMM-based HunPos tagger of Halacsy et al. (2007).

Ceaşu (2006) uses a maximum entropy Markov model (MEMM) based system for tagging Romanian which utilizes transitional behavior between sub-labels similarly to our feature set (6). However, in addition to ignoring the most in-

formative emission-type features (5), Ceaşu embeds the MEMMs into the tiered tagging framework of Tufis (1999). In tiered tagging, the full morphological analyses are mapped into a coarser tag set and a tagger is trained for this reduced tag set. Subsequent to decoding, the coarser tags are mapped into the original fine-grained morphological analyses. There are several problems associated with this tiered tagging approach. First, the success of the approach is highly dependent on a well designed coarse label set. Consequently, it requires intimate knowledge of the tag set and language. Meanwhile, our model can be set up with relatively little prior knowledge of the language or the tagging scheme (see Section 3.2). Moreover, a conversion to a coarser label set is necessarily lossy (at least for OOV words) and potentially results in reduced accuracy since recovering the original fine-grained tags from the coarse tags may induce errors. Indeed, the accuracy 96.56, reported by Ceaşu on the Romanian section of the Multext-East data set, is substantially lower than the accuracy 97.29 we obtain. These accuracies were obtained using identical sized training and test sets (although direct comparison is impossible because Ceaşu uses a non-documented random split).

5 Conclusions

We studied improving the accuracy of CRF-based POS tagging by exploiting sub-label dependency structure. The dependencies were included in the CRF model using a relatively straightforward feature expansion scheme. Experiments on five languages showed that the approach can yield significant improvement in tagging accuracy given sufficiently fine-grained label sets.

In future work, we aim to perform a more fine-grained error analysis to gain a better understanding where the improvement in accuracy takes place. One could also attempt to optimize the compound label splits to maximize prediction accuracy instead of applying a priori partitions.

Acknowledgements

This work was financially supported by Langnet (Finnish doctoral programme in language studies) and the Academy of Finland under the grant no 251170 (Finnish Centre of Excellence Program (2012-2017)). We would like to thank the anonymous reviewers for their useful comments.

References

- Thorsten Brants. 2000. Tnt: A statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 224–231.
- A. Ceausu. 2006. Maximum entropy tiered tagging. In *The 11th ESSLI Student session*, pages 173–179.
- Michael Collins. 2002. Discriminative training methods for Hidden Markov Models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, volume 10, pages 1–8.
- Tomaž Erjavec. 2010. Multext-east version 4: Multilingual morphosyntactic specifications, lexicons and corpora. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.
- Jan Hajič, Pavel Krbec, Pavel Květoň, Karel Oliva, and Vladimír Petkevič. 2001. Serial combination of rules and statistics: A case study in czech tagging. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 268–275.
- Auli Hakulinen, Maria Vilkuna, Riitta Korhonen, Vesa Koivisto, Tarja Riitta Heinonen, and Irja Alho. 2004. *Iso suomen kielioppi*. Suomalaisen Kirjallisuuden Seura, Helsinki, Finland.
- Péter Halácsy, András Kornai, and Csaba Oravecz. 2007. Hunpos: An open source trigram tagger. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, ACL '07*, pages 209–212.
- Katri Haverinen, Jenna Nyblom, Timo Viljanen, Veronika Laippala, Samuel Kohonen, Anna Missilä, Stina Ojala, Tapio Salakoski, and Filip Ginter. 2013. Building the essential resources for Finnish: the Turku Dependency Treebank. *Language Resources and Evaluation*.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.
- M.P. Marcus, M.A. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.
- Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, volume 1, pages 133–142. Philadelphia, PA.
- Noah A. Smith, David A. Smith, and Roy W. Tromble. 2005. Context-based morphological disambiguation with random fields. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 475–482.
- Dan Tufis. 1999. Tiered tagging and combined language models classifiers. In *Proceedings of the Second International Workshop on Text, Speech and Dialogue*, pages 28–33.
- Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1):105–151.

POS induction with distributional and morphological information using a distance-dependent Chinese restaurant process

Kairit Sirts

Institute of Cybernetics at
Tallinn University of Technology
sirts@ioc.ee

Jacob Eisenstein

School of Interactive Computing
Georgia Institute of Technology
jacobe@gatech.edu

Micha Elsner

Department of Linguistics
The Ohio State University
melsner0@gmail.com

Sharon Goldwater

ILCC, School of Informatics
University of Edinburgh
sgwater@inf.ed.ac.uk

Abstract

We present a new approach to inducing the syntactic categories of words, combining their distributional and morphological properties in a joint nonparametric Bayesian model based on the distance-dependent Chinese Restaurant Process. The prior distribution over word clusterings uses a log-linear model of morphological similarity; the likelihood function is the probability of generating vector word embeddings. The weights of the morphology model are learned jointly while inducing part-of-speech clusters, encouraging them to cohere with the distributional features. The resulting algorithm outperforms competitive alternatives on English POS induction.

1 Introduction

The morphosyntactic function of words is reflected in two ways: their distributional properties, and their morphological structure. Each information source has its own advantages and disadvantages. Distributional similarity varies smoothly with syntactic function, so that words with similar syntactic functions should have similar distributional properties. In contrast, there can be multiple paradigms for a single morphological inflection (such as past tense in English). But accurate computation of distributional similarity requires large amounts of data, which may not be available for rare words; morphological rules can be applied to any word regardless of how often it appears.

These observations suggest that a general approach to the induction of syntactic categories should leverage both distributional and morphological features (Clark, 2003; Christodoulopoulos

et al., 2010). But these features are difficult to combine because of their disparate representations. Distributional information is typically represented in numerical vectors, and recent work has demonstrated the utility of continuous vector representations, or “embeddings” (Mikolov et al., 2013; Luong et al., 2013; Kim and de Marneffe, 2013; Turian et al., 2010). In contrast, morphology is often represented in terms of sparse, discrete features (such as morphemes), or via pairwise measures such as string edit distance. Moreover, the mapping between a surface form and morphology is complex and nonlinear, so that simple metrics such as edit distance will only weakly approximate morphological similarity.

In this paper we present a new approach for inducing part-of-speech (POS) classes, combining morphological and distributional information in a non-parametric Bayesian generative model based on the *distance-dependent Chinese restaurant process* (ddCRP; Blei and Frazier, 2011). In the ddCRP, each data point (word type) selects another point to “follow”; this chain of following links corresponds to a partition of the data points into clusters. The probability of word w_1 following w_2 depends on two factors: 1) the *distributional* similarity between all words in the proposed partition containing w_1 and w_2 , which is encoded using a Gaussian likelihood function over the word embeddings; and 2) the *morphological* similarity between w_1 and w_2 , which acts as a prior distribution on the induced clustering. We use a log-linear model to capture suffix similarities between words, and learn the feature weights by iterating between sampling and weight learning.

We apply our model to the English section of the the Multext-East corpus (Erjavec, 2004) in order to evaluate both against the coarse-grained and

fine-grained tags, where the fine-grained tags encode detailed morphological classes. We find that our model effectively combines morphological features with distributional similarity, outperforming comparable alternative approaches.

2 Related work

Unsupervised POS tagging has a long history in NLP. This paper focuses on the POS induction problem (i.e., no tag dictionary is available), and here we limit our discussion to very recent systems. A review and comparison of older systems is provided by Christodoulopoulos et al. (2010), who found that imposing a one-tag-per-word-type constraint to reduce model flexibility tended to improve system performance; like other recent systems, we impose that constraint here. Recent work also shows that the combination of morphological and distributional information yields the best results, especially cross-linguistically (Clark, 2003; Berg-Kirkpatrick et al., 2010). Since then, most systems have incorporated morphology in some way, whether as an initial step to obtain prototypes for clusters (Abend et al., 2010), or as features in a generative model (Lee et al., 2010; Christodoulopoulos et al., 2011; Sirts and Alumäe, 2012), or a representation-learning algorithm (Yatbaz et al., 2012). Several of these systems use a small fixed set of orthographic and/or suffix features, sometimes obtained from an unsupervised morphological segmentation system (Abend et al., 2010; Lee et al., 2010; Christodoulopoulos et al., 2011; Yatbaz et al., 2012). Blunsom and Cohn’s (2011) model learns an n -gram character model over the words in each cluster; we learn a log-linear model, which can incorporate arbitrary features. Berg-Kirkpatrick et al. (2010) also include a log-linear model of morphology in POS induction, but they use morphology in the likelihood term of a parametric sequence model, thereby encouraging all elements that share a tag to have the same morphological features. In contrast, we use *pairwise morphological similarity* as a prior in a non-parametric clustering model. This means that the membership of a word in a cluster requires only morphological similarity to some other element in the cluster, not to the cluster centroid; which may be more appropriate for languages with multiple morphological paradigms. Another difference is that our non-parametric formulation makes it unnecessary to know the number of tags in advance.

3 Distance-dependent CRP

The ddCRP (Blei and Frazier, 2011) is an extension of the CRP; like the CRP, it defines a distribution over partitions (“table assignments”) of data points (“customers”). Whereas in the regular CRP each customer chooses a table with probability proportional to the number of customers already sitting there, in the ddCRP each customer chooses another *customer* to follow, and sits at the same table with that customer. By identifying the connected components in this graph, the ddCRP equivalently defines a prior over clusterings.

If c_i is the index of the customer followed by customer i , then the ddCRP prior can be written

$$P(c_i = j) \propto \begin{cases} f(d_{ij}) & \text{if } i \neq j \\ \alpha & \text{if } i = j, \end{cases} \quad (1)$$

where d_{ij} is the distance between customers i and j and f is a decay function. A ddCRP is *sequential* if customers can only follow previous customers, i.e., $d_{ij} = \infty$ when $i > j$ and $f(\infty) = 0$. In this case, if $d_{ij} = 1$ for all $i < j$ then the ddCRP reduces to the CRP.

Separating the distance and decay function makes sense for “natural” distances (e.g., the number of words between word i and j in a document, or the time between two events), but they can also be collapsed into a single similarity function. We wish to assign higher similarities to pairs of words that share meaningful suffixes. Because we do not know which suffixes are meaningful *a priori*, we use a maximum entropy model whose features include all suffixes up to length three that are shared by at least one pair of words. Our prior is then:

$$P(c_i = j | \mathbf{w}, \alpha) \propto \begin{cases} e^{\mathbf{w}^\top \mathbf{g}^{(i,j)}} & \text{if } i \neq j \\ \alpha & \text{if } i = j, \end{cases} \quad (2)$$

where $g_s(i, j)$ is 1 if suffix s is shared by i th and j th words, and 0 otherwise.

We can create an infinite mixture model by combining the ddCRP prior with a likelihood function defining the probability of the data given the cluster assignments. Since we are using continuous-valued vectors (word embeddings) to represent the distributional characteristics of words, we use a multivariate Gaussian likelihood. We will marginalize over the mean μ and covariance Σ of each cluster, which in turn are drawn from Gaussian and inverse-Wishart (IW) priors respectively:

$$\Sigma \sim IW(\nu_0, \Lambda_0) \quad \mu \sim \mathcal{N}(\mu_0, \Sigma / \kappa_0) \quad (3)$$

The full model is then:

$$\begin{aligned}
P(\mathbf{X}, \mathbf{c}, \boldsymbol{\mu}, \boldsymbol{\Sigma} | \Theta, \mathbf{w}, \alpha) & \quad (4) \\
&= \prod_{k=1}^K P(\Sigma_k | \Theta) p(\boldsymbol{\mu}_k | \Sigma_k, \Theta) \\
&\quad \times \prod_{i=1}^n (P(c_i | \mathbf{w}, \alpha) P(\mathbf{x}_i | \boldsymbol{\mu}_{z_i}, \Sigma_{z_i})),
\end{aligned}$$

where Θ are the hyperparameters for $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and z_i is the (implicit) cluster assignment of the i th word \mathbf{x}_i . With a CRP prior, this model would be an infinite Gaussian mixture model (IGMM; Rasmussen, 2000), and we will use the IGMM as a baseline.

4 Inference

The Gibbs sampler for the ddCRP integrates over the Gaussian parameters, sampling only follower variables. At each step, the follower link c_i for a single customer i is sampled, which can implicitly shift the entire block of n customers $\text{fol}(i)$ who follow i into a new cluster. Since we marginalize over the cluster parameters, computing $P(c_i = j)$ requires computing the likelihood $P(\text{fol}(i), \mathbf{X}_j | \Theta)$, where \mathbf{X}_j are the k customers already clustered with j . However, if we do *not* merge $\text{fol}(i)$ with \mathbf{X}_j , then we have $P(\mathbf{X}_j | \Theta)$ in the overall joint probability. Therefore, we can decompose $P(\text{fol}(i), \mathbf{X}_j | \Theta) = P(\text{fol}(i) | \mathbf{X}_j, \Theta) P(\mathbf{X}_j | \Theta)$ and need only compute the change in likelihood due to merging in $\text{fol}(i)$:¹

$$\begin{aligned}
P(\text{fol}(i) | \mathbf{X}_j, \Theta) &= \pi^{-nd/2} \frac{\kappa_k^{d/2} |\Lambda_k|^{\nu_k/2}}{\kappa_{n+k}^{d/2} |\Lambda_{n+k}|^{\nu_{n+k}/2}} \\
&\quad \times \prod_{i=1}^d \frac{\Gamma\left(\frac{\nu_{n+k}+1-i}{2}\right)}{\Gamma\left(\frac{\nu_k+1-i}{2}\right)}, \quad (5)
\end{aligned}$$

where the hyperparameters are updated as $\kappa_n = \kappa_0 + n$, $\nu_n = \nu_0 + n$, and

$$\mu_n = \frac{\kappa_0 \mu_0 + \bar{x}}{\kappa_0 + n} \quad (6)$$

$$\Lambda_n = \Lambda_0 + Q + \kappa_0 \mu_0 \mu_0^T - \kappa_n \mu_n \mu_n^T, \quad (7)$$

where $Q = \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^T$.

Combining this likelihood term with the prior, the probability of customer i following j is

$$\begin{aligned}
P(c_i = j | \mathbf{X}, \Theta, \mathbf{w}, \alpha) \\
\propto P(\text{fol}(i) | \mathbf{X}_j, \Theta) P(c_i = j | \mathbf{w}, \alpha). \quad (8)
\end{aligned}$$

¹<http://www.stats.ox.ac.uk/~teh/research/notes/GaussianInverseWishart.pdf>

Our non-sequential ddCRP introduces cycles into the follower structure, which are handled in the sampler as described by Socher et al. (2011). Also, the block of customers being moved around can potentially be very large, which makes it easy for the likelihood term to swamp the prior. In practice we found that introducing an additional parameter a (used to exponentiate the prior) improved results—although we report results without this exponent as well. This technique was also used by Titov and Klementiev (2012) and Elsner et al. (2012).

Inference also includes optimizing the feature weights for the log-linear model in the ddCRP prior (Titov and Klementiev, 2012). We interleave L-BFGS optimization within sampling, as in Monte Carlo Expectation-Maximization (Wei and Tanner, 1990). We do not apply the exponentiation parameter a when training the weights because this procedure affects the follower structure only, and we do not have to worry about the magnitude of the likelihood. Before the first iteration we initialize the follower structure: for each word, we choose randomly a word to follow from amongst those with the longest shared suffix of up to 3 characters. The number of clusters starts around 750, but decreases substantially after the first sampling iteration.

5 Experiments

Data For our experiments we used the English word embeddings from the Polyglot project (Al-Rfou’ et al., 2013)², which provides embeddings trained on Wikipedia texts for 100,000 of the most frequent words in many languages.

We evaluate on the English part of the Multext-East (MTE) corpus (Erjavec, 2004), which provides both coarse-grained and fine-grained POS labels for the text of Orwell’s “1984”. Coarse labels consist of 11 main word classes, while the fine-grained tags (104 for English) are sequences of detailed morphological attributes. Some of these attributes are not well-attested in English (e.g. gender) and some are mostly distinguishable via semantic analysis (e.g. 1st and 2nd person verbs). Many tags are assigned only to one or a few words. Scores for the fine-grained tags will be lower for these reasons, but we argue below that they are still informative.

Since Wikipedia and MTE are from different domains their lexicons do not fully overlap; we

²<https://sites.google.com/site/rmyeid/projects/polyglot>

Wikipedia tokens	1843M
Multext-East tokens	118K
Multext-East types	9193
Multext-East & Wiki types	7540

Table 1: Statistics for the English Polyglot word embeddings and English part of MTE: number of Wikipedia tokens used to train the embeddings, number of tokens/types in MTE, and number of types shared by both datasets.

take the intersection of these two sets for training and evaluation. Table 1 shows corpus statistics.

Evaluation With a few exceptions (Biemann, 2006; Van Gael et al., 2009), POS induction systems normally require the user to specify the number of desired clusters, and the systems are evaluated with that number set to the number of tags in the gold standard. For corpora such as MTE with both fine-grained and coarse-grained tags, previous evaluations have scored against the coarse-grained tags. Though coarse-grained tags have their place (Petrov et al., 2012), in many cases the distributional and morphological distinctions between words are more closely aligned with the fine-grained tagsets, which typically distinguish between verb tenses, noun number and gender, and adjectival scale (comparative, superlative, etc.), so we feel that the evaluation against fine-grained tagset is more relevant here. For better comparison with previous work, we also evaluate against the coarse-grained tags; however, these numbers are not strictly comparable to other scores reported on MTE because we are only able to train and evaluate on the subset of words that also have Polyglot embeddings. To provide some measure of the difficulty of the task, we report baseline scores using K-means clustering, which is relatively strong baseline in this task (Christodoulopoulos et al., 2011).

There are several measures commonly used for unsupervised POS induction. We report greedy one-to-one mapping accuracy (1-1) (Haghighi and Klein, 2006) and the information-theoretic score V-measure (V-m), which also varies from 0 to 100% (Rosenberg and Hirschberg, 2007). In previous work it has been common to also report many-to-one (m-1) mapping but this measure is particularly sensitive to the number of induced clusters (more clusters yield higher scores), which is variable for our models. V-m can be somewhat sensitive to the number of clusters (Reichart and Rappoport, 2009) but much less so than m-1 (Christodoulopoulos

et al., 2010). With different number of induced and gold standard clusters the 1-1 measure suffers because some induced clusters cannot be mapped to gold clusters or vice versa. However, almost half the gold standard clusters in MTE contain just a few words and we do not expect our model to be able to learn them anyway, so the 1-1 measure is still useful for telling us how well the model learns the bigger and more distinguishable classes.

In unsupervised POS induction it is standard to report accuracy on tokens even when the model itself works on types. Here we report also type-based measures because these can reveal differences in model behavior even when token-based measures are similar.

Experimental setup For baselines we use K-means and the IGMM, which both only learn from the word embeddings. The CRP prior in the IGMM has one hyperparameter (the concentration parameter α); we report results for $\alpha = 5$ and 20. Both the IGMM and ddCRP have four hyperparameters controlling the prior over the Gaussian cluster parameters: Λ_0 , μ_0 , ν_0 and κ_0 . We set the prior scale matrix Λ_0 by using the average covariance from a K-means run with $K = 200$. When setting the average covariance as the expected value of the IW distribution the suitable scale matrix can be computed as $\Lambda_0 = E[X](\nu_0 - d - 1)$, where ν_0 is the prior degrees of freedom (which we set to $d + 10$) and d is the data dimensionality (64 for the Polyglot embeddings). We set the prior mean μ_0 equal to the sample mean of the data and κ_0 to 0.01.

We experiment with three different priors for the ddCRP model. All our ddCRP models are non-sequential (Socher et al., 2011), allowing cycles to be formed. The simplest model, *ddCRP uniform*, uses a uniform prior that sets the distance between any two words equal to one.³ The second model, *ddCRP learned*, uses the log-linear prior with weights learned between each two Gibbs iterations as explained in section 4. The final model, *ddCRP exp*, adds the prior exponentiation. The α parameter for the ddCRP is set to 1 in all experiments. For *ddCRP exp*, we report results with the exponent a set to 5.

Results and discussion Table 2 presents all results. Each number is an average of 5 experiments

³In the sequential case this model would be equivalent to the IGMM (Blei and Frazier, 2011). Due to the nonsequentiality this equivalence does not hold, but we do expect to see similar results to the IGMM.

Model	K	Fine types		Fine tokens		Coarse tokens	
		Model	K-means	Model	K-means	Model	K-means
K-means	104 or 11	16.1 / 47.3	-	39.2 / 62.0	-	44.4 / 45.5	-
IGMM, $\alpha = 5$	55.6	41.0 / 45.9	23.1 / 49.5	48.0 / 64.8	37.2 / 61.0	48.3 / 58.3	40.8 / 55.0
IGMM, $\alpha = 20$	121.2	35.0 / 47.1	14.7 / 46.9	50.6 / 67.8	44.7 / 65.5	48.7 / 60.0	48.3 / 57.9
ddCRP uniform	80.4	50.5 / 52.9	18.6 / 48.2	52.4 / 68.7	35.1 / 60.3	52.1 / 62.2	40.3 / 54.2
ddCRP learned	89.6	50.1 / 55.1	17.6 / 48.0	51.1 / 69.7	39.0 / 63.2	48.9 / 62.0	41.1 / 55.1
ddCRP exp, $a = 5$	47.2	64.0 / 60.3	25.0 / 50.3	55.1 / 66.4	33.0 / 59.1	47.8 / 55.1	36.9 / 53.1

Table 2: Results of baseline and ddCRP models evaluated on word types and tokens using fine-grained tags, and on tokens using coarse-grained tags. For each model we present the number of induced clusters K (or fixed K for K-means) and 1-1 / V-m scores. The second column under each evaluation setting gives the scores for K-means with K equal to the number of clusters induced by the model in that row.

with different random initializations. For each evaluation setting we provide two sets of scores—first are the 1-1 and V-m scores for the given model, second are the comparable scores for K-means run with the same number of clusters as induced by the non-parametric model.

These results show that all non-parametric models perform better than K-means, which is a strong baseline in this task (Christodoulopoulos et al., 2011). The poor performance of K-means can be explained by the fact that it tends to find clusters of relatively equal size, although the POS clusters are rarely of similar size. The common noun singular class is by far the largest in English, containing roughly a quarter of the word types. Non-parametric models are able to produce cluster of different sizes when the evidence indicates so, and this is clearly the case here.

From the token-based evaluation it is hard to say which IGMM hyperparameter value is better even though the number of clusters induced differs by a factor of 2. The type-base evaluation, however, clearly prefers the smaller value with fewer clusters. Similar effects can be seen when comparing IGMM and ddCRP uniform. We expected these two models perform on the same level, and their token-based scores are similar, but on the type-based evaluation the ddCRP is clearly superior. The difference could be due to the non-sequentiality, or because the samplers are different—IGMM enabling resampling only one item at a time, ddCRP performing blocked sampling.

Further we can see that the ddCRP uniform and learned perform roughly the same. Although the prior in those models is different they work mainly using the the likelihood. The ddCRP with learned prior does produce nice follower structures within each cluster but the prior is in general too weak compared to the likelihood to influence the clustering decisions. Exponentiating the prior reduces the

number of induced clusters and improves results, as it can change the cluster assignment for some words where the likelihood strongly prefers one cluster but the prior clearly indicates another.

The last column shows the token-based evaluation against the coarse-grained tagset. This is the most common evaluation framework used previously in the literature. Although our scores are not directly comparable with the previous results, our V-m scores are similar to the best published 60.5 (Christodoulopoulos et al., 2010) and 66.7 (Sirts and Alumäe, 2012).

In preliminary experiments, we found that directly applying the best-performing English model to other languages is not effective. Different languages may require different parametrizations of the model. Further study is also needed to verify that word embeddings effectively capture syntax across languages, and to determine the amount of unlabeled text necessary to learn good embeddings.

6 Conclusion

This paper demonstrates that morphology and distributional features can be combined in a flexible, joint probabilistic model, using the distance-dependent Chinese Restaurant Process. A key advantage of this framework is the ability to include arbitrary features in the prior distribution. Future work may exploit this advantage more thoroughly: for example, by using features that incorporate prior knowledge of the language’s morphological structure. Another important goal is the evaluation of this method on languages beyond English.

Acknowledgments: KS was supported by the Tiger University program of the Estonian Information Technology Foundation for Education. JE was supported by a visiting fellowship from the Scottish Informatics & Computer Science Alliance. We thank the reviewers for their helpful feedback.

References

- Omri Abend, Roi Reichart, and Ari Rappoport. 2010. Improved unsupervised pos induction through prototype discovery. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics*, pages 1298–1307.
- Rami Al-Rfou’, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual nlp. In *Proceedings of the Thirteenth Annual Conference on Natural Language Learning*, pages 183–192, Sofia, Bulgaria. Association for Computational Linguistics.
- Taylor Berg-Kirkpatrick, Alexandre B. Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 582–590.
- Chris Biemann. 2006. Unsupervised part-of-speech tagging employing efficient graph clustering. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 7–12.
- David M Blei and Peter I Frazier. 2011. Distance dependent chinese restaurant processes. *Journal of Machine Learning Research*, 12:2461–2488.
- Phil Blunsom and Trevor Cohn. 2011. A hierarchical pitman-yor process hmm for unsupervised part of speech induction. In *Proceedings of the 49th Annual Meeting of the Association of Computational Linguistics*, pages 865–874.
- Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2010. Two decades of unsupervised POS induction: How far have we come? In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2011. A Bayesian mixture model for part-of-speech induction using multiple features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Alexander Clark. 2003. Combining distributional and morphological information for part of speech induction. In *Proceedings of the European chapter of the ACL*.
- Micha Elsner, Sharon Goldwater, and Jacob Eisenstein. 2012. Bootstrapping a unified model of lexical and phonetic acquisition. In *Proceedings of the 50th Annual Meeting of the Association of Computational Linguistics*.
- Tomaž Erjavec. 2004. MULTEXT-East version 3: Multilingual morphosyntactic specifications, lexicons and corpora. In *LREC*.
- A. Haghighi and D. Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*.
- Joo-Kyung Kim and Marie-Catherine de Marneffe. 2013. Deriving adjectival scales from continuous space word representations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. 2010. Simple type-level unsupervised pos tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 853–861.
- Minh-Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Thirteenth Annual Conference on Natural Language Learning*.
- Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 746–751.
- Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of LREC*, May.
- Carl Rasmussen. 2000. The infinite Gaussian mixture model. In *Advances in Neural Information Processing Systems 12*, Cambridge, MA. MIT Press.
- Roi Reichart and Ari Rappoport. 2009. The nvi clustering evaluation measure. In *Proceedings of the Ninth Annual Conference on Natural Language Learning*, pages 165–173.
- A. Rosenberg and J. Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 410–42.
- Kairit Sirts and Tanel Alumäe. 2012. A hierarchical Dirichlet process model for joint part-of-speech and morphology induction. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 407–416.
- Richard Socher, Andrew L Maas, and Christopher D Manning. 2011. Spectral chinese restaurant processes: Nonparametric clustering based on similarities. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, pages 698–706.

- Ivan Titov and Alexandre Klementiev. 2012. A bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*.
- Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden, July. Association for Computational Linguistics.
- Jurgen Van Gael, Andreas Vlachos, and Zoubin Ghahramani. 2009. The infinite HMM for unsupervised PoS tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 678–687, Singapore.
- Greg CG Wei and Martin A Tanner. 1990. A monte carlo implementation of the em algorithm and the poor man’s data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704.
- Mehmet Ali Yatbaz, Enis Sert, and Deniz Yuret. 2012. Learning syntactic categories using paradigmatic representations of word context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 940–951.

Improving the Recognizability of Syntactic Relations Using Contextualized Examples

Aditi Muralidharan

Computer Science Division
University of California, Berkeley
Berkeley, CA
asm@berkeley.edu

Marti A. Hearst

School of Information
University of California, Berkeley
Berkeley, CA
hearst@berkeley.edu

Abstract

A common task in qualitative data analysis is to characterize the usage of a linguistic entity by issuing queries over syntactic relations between words. Previous interfaces for searching over syntactic structures require programming-style queries. User interface research suggests that it is easier to recognize a pattern than to compose it from scratch; therefore, interfaces for non-experts should show previews of syntactic relations. What these previews should look like is an open question that we explored with a 400-participant Mechanical Turk experiment. We found that syntactic relations are recognized with 34% higher accuracy when contextual examples are shown than a baseline of naming the relations alone. This suggests that user interfaces should display contextual examples of syntactic relations to help users choose between different relations.

1 Introduction

The ability to search over grammatical relationships between words is useful in many non-scientific fields. For example, a social scientist trying to characterize different perspectives on immigration might ask how adjectives applying to ‘immigrant’ have changed in the last 30 years. A scholar interested in gender might search a collection to find out whether different nouns enter into possessive relationships with ‘his’ and ‘her’ (Muralidharan and Hearst, 2013). In other fields, grammatical queries can be used to develop patterns for recognizing entities in text, such as medical terms (Hirschman et al., 2005; MacLean and Heer, 2013), and products and organizations (Culotta and McCallum, 2005), and for coding qualitative data such as survey results.

Most existing interfaces for syntactic search (querying over grammatical and syntactic structures) require structured query syntax. For example, the popular Stanford Parser includes Tregex, which allows for sophisticated regular expression search over syntactic tree structures (Levy and Andrew, 2006). The Finite Structure Query tool for querying syntactically annotated corpora requires its queries to be stated in first order logic (Kepser, 2003). In the Corpus Query Language (Jakubicek et al., 2010), a query is a pattern of attribute-value pairs, where values can include regular expressions containing parse tree nodes and words. Several approaches have adopted XML representations and the associated query language families of XPATH and SPARQL. For example, LPath augments XPath with additional tree operators to give it further expressiveness (Lai and Bird, 2010).

However, most potential users do not have programming expertise, and are not likely to be at ease composing rigidly-structured queries. One survey found that even though linguists wished to make very technical linguistic queries, 55% of them did not know how to program (Soehn et al., 2008). In another (Gibbs and Owens, 2012), humanities scholars and social scientists are frequently skeptical of digital tools, because they are often difficult to use. This reduces the likelihood that existing structured-query tools for syntactic search will be usable by non-programmers (Ogden and Brooks, 1983).

A related approach is the query-by-example work seen in the past in interfaces to database systems (Androutsopoulos et al., 1995). For instance, the Linguist’s Search Engine (Resnik et al., 2005) uses a query-by-example strategy in which a user types in an initial sentence in English, and the system produces a graphical view of a parse tree as output, which the user can alter. The user can either click on the tree or modify the LISP expression to generalize the query. SPLICR also contains

a graphical tree editor tool (Rehm et al., 2009). According to Shneiderman and Plaisant (2010), query-by-example has largely fallen out of favor as a user interface design approach. A downside of QBE is that the user must manipulate an example to arrive at the desired generalization.

More recently auto-suggest, a faster technique that does not require the manipulation of query by example, has become a widely-used approach in search user interfaces with strong support in terms of its usability (Anick and Kantamneni, 2008; Ward et al., 2012; Jagadish et al., 2007). A list of selectable options is shown under the search bar, filtered to be relevant as the searcher types. Searchers can recognize and select the option that matches their information need, without having to generate the query themselves.

The success of auto-suggest depends upon showing users options they can recognize. However, we know of no prior work on how to display grammatical relations so that they can be easily recognized. One current presentation (not used with auto-suggest) is to name the relation and show blanks where the words that satisfy it would appear as in *X is the subject of Y* (Muralidharan and Hearst, 2013); we used this as the baseline presentation in our experiments because it employs the relation definitions found in the Stanford Dependency Parser’s manual (De Marneffe et al., 2006). Following the principle of recognition over recall, we hypothesized that showing contextualized usage examples would make the relations more recognizable.

Our results confirm that showing examples in the form of words or phrases significantly improves the accuracy with which grammatical relationships are recognized over the standard baseline of showing the relation name with blanks. Our findings also showed that clausal relationships, which span longer distances in sentences, benefited significantly more from example phrases than either of the other treatments.

These findings suggest that a query interface in which a user enters a word of interest and the system shows candidate grammatical relations augmented with examples from the text will be more successful than the baseline of simply naming the relation and showing gaps where the participating words appear.

2 Experiment

We gave participants a series of identification tasks. In each task, they were shown a list of sentences containing a particular syntactic relationship between highlighted words. They were asked to identify the relationship type from a list of four options. We presented the options in three different ways, and compared the accuracy.

We chose Amazon’s Mechanical Turk (MTurk) crowdsourcing platform as a source of study participants. The wide range of backgrounds provided by MTurk is desirable because our goal is to find a representation that is understandable to most people, not just linguistic experts or programmers. This platform has become widely used for both obtaining language judgements and for usability studies (Kittur et al., 2008; Snow et al., 2008).

Our hypothesis was:

Grammatical relations are identified more accurately when shown with examples of contextualizing words or phrases than without.

To test it, participants were given a series of identification tasks. In each task, they were shown a list of 8 sentences, each containing a particular relationship between highlighted words. They were asked to identify the relationship from a list of 4 choices. Additionally, one word was chosen as a *focus word* that was present in all the sentences, to make the relationship more recognizable (“life” in Figure 1).

The choices were displayed in 3 different ways (Figure 1). The **baseline** presentation (Figure 1a) named the linguistic relation and showed a blank space with a pink background for the varying word in the relationship, the focus word highlighted in yellow and underlined, and any necessary additional words necessary to convey the relationship (such as “of” for the prepositional relationship “of”, the third option).

The **words** presentation showed the baseline design, and in addition beneath was the word “Examples:” followed by a list of 4 example words that could fill in the pink blank slot (Figure 1b). The **phrases** presentation again showed the baseline design, beneath which was the phrase “Patterns like:” and a list of 4 example phrases in which fragments of text including both the pink and the yellow highlighted portions of the relationship appeared (Figure 1c).

possessive: **life**

determiner: **life**

preposition: **life** of

adjective modifier: **life**

possessive: **life**
Examples: **my**, **Woodhouse**, **her**, **his** etc.

determiner: **life**
Examples: **this**, **that**, **an**, **no** etc.

preposition: **life** of
Examples: **whale**, **Canal**, **them**, **Job** etc.

adjective modifier: **life**
Examples: **common**, **seagoing**, **tropic**, **often** etc.

(a) The options as they appear in the *baseline* condition.

(b) The same options as they appear in the *words* condition.

Choose the option that best describes the grammatical relationship between the highlighted words in the sentences on the right.

possessive: **life**
Patterns like:
• place in **my life**.
• of Mr. Woodhouse's **life**; and
• never in **her life** been within
• part of his **life**, and

determiner: **life**
Patterns like:
• wildness of **this canal life** is,
• , whether that **invaluable life** of his
• his is an **unwritten life**.
• There is **no life** in thee

preposition: **life** of
Patterns like:
• the fathom-deep **life** of the whale.
• the probationary **life** of the Grand Canal furnishes the
• for the **life** of them, can
• in the **life** of patient Job.

adjective modifier: **life**
Patterns like:
• because in **common life** we esteem
• sort of **seagoing life**, in
• in this **tropic whaling life**, a
• corrupt and often **lawless life**.

And as the sea surpasses the land in this matter, so the whale fishery surpasses every other sort of **maritime life**, in the wonderfulness and fearfulness of the rumors which sometimes circulate there.

So Tamerlane's soldiers often argued with tears in their eyes, whether that **invaluable life** of his ought to be carried into the thickest of the fight.

" Now, the Captain D'Wolf here alluded to as commanding the ship in question, is a New Englander, who, after a **long life** of unusual adventures as a sea-captain, this day resides in the village of Dorchester near Boston.

And what with the standing spectacle of the black terrific Ahab, and the periodical tumultuous visitations of these three savages, Dough-Boy's **whole life** was one continual lip-quiver.

Men, ye seem the years; so **brimming life** is gulped and gone.

No one having previously heard his history, could for the first time behold Father Mapple without the utmost interest, because there were certain engrafted clerical peculiarities about him, imputable to that **adventurous maritime life** he had led.

It seemed as though, by some nameless, interior volition, he would fain have shocked into them the same fiery emotion accumulated within the Leyden jar of his own **magnetic life**.

He would say the most terrific things to his crew, in a tone so strangely compounded of fun and fury, and the fury seemed so calculated merely as a spice to the fun, that no oarsman could hear such queer invocations without pulling for **dear life**, and yet pulling for the mere joke of the thing.

(c) The same options in the *phrases* condition, shown as they appeared in an identification task for the relationship amod(life, ...) (where different adjectives modify the noun 'life'). The correct answer is 'adjective modifier' (4th option), and the remaining 3 options are distractors.

Figure 1: The appearance of the choices shown in the three experiment conditions.

Method: We used a between-subjects design. The task order and the choice order were not varied: the only variation between participants was the presentation of the choices. To avoid the possibility of guessing the right answer by pattern-matching, we ensured that there was no overlap between the list of sentences shown, and the examples shown in the choices as words or phrases.

Tasks: The tasks were generated using the Stanford Dependency Parser (De Marneffe et al., 2006) on the text of *Moby Dick* by Herman Melville. We tested the 12 most common grammatical relationships in the novel in order to cover the most content and to be able to provide as many real examples as possible. These relationships fall

into two categories, listed below with examples.

Clausal or long-distance relations:

- Adverbial clause: *I walk while talking*
- Open clausal complement: *I love to sing*
- Clausal complement: *he saw us leave*
- Relative clause modifier: *the letter I wrote reached*

Non-clausal relations:

- Subject of verb: *he threw the ball*
- Object of verb: *he threw the ball*
- Adjective modifier *red ball*
- Preposition (in): *a hole in a bucket*
- Preposition (of): *the piece of cheese*
- Conjunction (and) *mind and body*

- Adverb modifier: *we walk slowly*
- Noun compound: *Mr. Brown*

We tested each of these 12 relations with 4 different focus words, 2 in each role. For example, the *Subject of Verb* relation was tested in the following forms:

- (*Ahab*, ___): the sentences each contained ‘Ahab’, highlighted in yellow, as the subject of different verbs highlighted in pink.
- (*captain*, ___)
- (___, *said*): the sentences each contained the verb ‘said’, highlighted in yellow, but with different subjects, highlighted in pink.
- (___, *stood*)

To maximize coverage, yet keep the total task time reasonable (average 6.8 minutes), we divided the relations above into 4 task sets, each testing recognition of 3 different relations. Each of relations was tested with 4 different words, making a total of 12 tasks per participant.

Participants: 400 participants completed the study distributed randomly over the 4 task sets and the 3 presentations. Participants were paid 50c (U.S.) for completing the study, with an additional 50c bonus if they correctly identified 10 or more of the 12 relationships. They were informed of the possibility of the bonus before starting.

To gauge their syntactic familiarity, we also asked them to rate how familiar they were with the terms ‘adjective’ (88% claimed they could define it), ‘infinitive’ (43%), and ‘clausal complement’ (18%). To help ensure the quality of effort, we included a multiple-choice screening question, “What is the third word of this sentence?” The 27 participants (out of 410) who answered incorrectly were eliminated.

Results: The results (Figure 2) confirm our hypothesis. Participants in conditions that showed examples (**phrases** and **words**) were significantly more accurate at identifying the relations than participants in the **baseline** condition. We used the Wilcoxon signed-rank test, an alternative to the standard T-test that does not assume samples are normally distributed. The average success rate in the **baseline** condition was 41%, which is significantly less accurate than **words**: 52%, ($p=0.00019$, $W=6136$), and **phrases**: 55%, ($p=0.00014$, $W=5546.5$).

Clausal relations operate over longer distances in sentences, and so it is to be expected that showing longer stretches of context would perform bet-

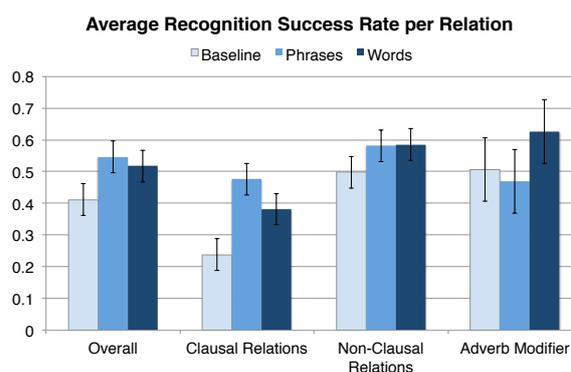


Figure 2: Recognition rates for different types of relations under the 3 experiment conditions, with 95% confidence intervals.

ter in these cases; that is indeed what the results showed. Phrases significantly outperformed words and baseline for clausal relations. The average success rate was 48% for **phrases**, which is significantly more than **words**: 38%, ($p=0.017$ $W=6976.5$) and **baseline**: 24%, ($p=1.9 \times 10^{-9}$ $W=4399.0$), which was indistinguishable from random guessing (25%). This is a strong improvement, given that only 18% of participants reported being able to define ‘clausal complement’.

For the non-clausal relations, there was no significant difference between **phrases** and **words**, although they were both overall significantly better than the baseline (words: $p=0.0063$ $W=6740$, phrases: $p=0.023$ $W=6418.5$). Among these relations, adverb modifiers stood out (Figure 2), because evidence suggested that **words** (63% success) made the relation more recognizable than **phrases** (47% success, $p=0.056$, $W=574.0$) – but the difference was only almost significant, due to the smaller sample size (only 96 participants encountered this relation). This may be because the words are the most salient piece of information in an adverbial relation – adverbs usually end in ‘ly’ – and in the phrases condition the additional information distracts from recognition of this pattern.

3 Conclusions

The results imply that user interfaces for syntactic search should show candidate relationships augmented with a list of phrases in which they occur. A list of phrases is the most recognizable presentation for clausal relationships (34% better than the baseline), and is as good as a list of words for the other types of relations, except adverb modifiers. For adverb modifiers, the list of words is the most

recognizable presentation. This is likely because English adverbs usually end in ‘-ly’ are therefore a distinctive set of words.

The list of candidates can be ordered by frequency of occurrence in the collection, or by an interestingness measure given the search word. As the user becomes more familiar with a given relation, it may be expedient to shorten the cues shown, and then re-introduce them if a relation has not been selected after some period of time has elapsed. If phrases are used, there is a tradeoff between recognizability and the space required to display the examples of usage. However, it is important to keep in mind that because the suggestions are populated with items from the collection itself, they are informative.

The best strategy, **phrases**, had an overall success rate of only 55%, although the intended user base may have more familiarity with grammatical relations than the participants did, and therefore may perform better in practice. Nonetheless, there is room for improvement in scores, and it may be that additional visual cues, such as some kind of bracketing, will improve results. Furthermore, the current study did not test three-word relationships or more complex combinations of structures, and those may require improvements to the design.

4 Acknowledgements

We thank Björn Hartmann for his helpful comments. This work is supported by National Endowment for the Humanities grant HK-50011.

References

- I Androutsopoulos, GD Ritchie, and P Thanisch. 1995. Natural language interfaces to databases—an introduction. *Natural Language Engineering*, 1(01):29–81.
- Peter Anick and Raj Gopal Kantamneni. 2008. A longitudinal study of real-time search assistance adoption. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 701–702. ACM.
- Aron Culotta and Andrew McCallum. 2005. Reducing labeling effort for structured prediction tasks. In *AAAI*, pages 746–751.
- Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *LREC*, volume 6, pages 449–454.
- Fred Gibbs and Trevor Owens. 2012. Building better digital humanities tools. *DH Quarterly*, 6(2).
- Lynette Hirschman, Alexander Yeh, Christian Blaschke, and Alfonso Valencia. 2005. Overview of biocreative: critical assessment of information extraction for biology. *BMC bioinformatics*, 6(Suppl 1):S1.
- HV Jagadish, Adriane Chapman, Aaron Elkiss, Magesh Jayapandian, Yunyao Li, Arnab Nandi, and Cong Yu. 2007. Making database systems usable. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 13–24. ACM.
- Milos Jakubicek, Adam Kilgarriff, Diana McCarthy, and Pavel Rychlý. 2010. Fast syntactic searching in very large corpora for many languages. In *PACLIC*, volume 24, pages 741–747.
- Stephan Kepser. 2003. Finite structure query: A tool for querying syntactically annotated corpora. In *EACL*, pages 179–186.
- Aniket Kittur, Ed H Chi, and Bongwon Suh. 2008. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–456. ACM.
- Catherine Lai and Steven Bird. 2010. Querying linguistic trees. *Journal of Logic, Language and Information*, 19(1):53–73.
- Roger Levy and Galen Andrew. 2006. Tregex and tsurgeon: tools for querying and manipulating tree data structures. In *LREC*, pages 2231–2234.
- Diana Lynn MacLean and Jeffrey Heer. 2013. Identifying medical terms in patient-authored text: a crowdsourcing-based approach. *Journal of the American Medical Informatics Association*.
- Aditi Muralidharan and Marti A Hearst. 2013. Supporting exploratory text analysis in literature study. *Literary and Linguistic Computing*, 28(2):283–295.
- William C Ogden and Susan R Brooks. 1983. Query languages for the casual user: Exploring the middle ground between formal and natural languages. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 161–165. ACM.
- Georg Rehm, Oliver Schonefeld, Andreas Witt, Erhard Hinrichs, and Marga Reis. 2009. Sustainability of annotated resources in linguistics: A web-platform for exploring, querying, and distributing linguistic corpora and other resources. *Literary and Linguistic Computing*, 24(2):193–210.
- Philip Resnik, Aaron Elkiss, Ellen Lau, and Heather Taylor. 2005. The web in theoretical linguistics research: Two case studies using the linguists search engine. In *Proc. 31st Mtg. Berkeley Linguistics Society*, pages 265–276.

Ben Shneiderman and Catherine Plaisant. 2010. *Designing The User Interface: Strategies for Effective Human-Computer Interaction, 5/e (Fifth Edition)*. Addison Wesley.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 254–263. Association for Computational Linguistics.

Jan-Philipp Soehn, Heike Zinsmeister, and Georg Rehm. 2008. Requirements of a user-friendly, general-purpose corpus query interface. *Sustainability of Language Resources and Tools for Natural Language Processing*, 6:27.

David Ward, Jim Hahn, and Kirsten Feist. 2012. Autocomplete as research tool: A study on providing search suggestions. *Information Technology and Libraries*, 31(4):6–19.

How to Speak a Language without Knowing It

Xing Shi and Kevin Knight

Information Sciences Institute
Computer Science Department
University of Southern California
{xingshi, knight}@isi.edu

Heng Ji

Computer Science Department
Rensselaer Polytechnic Institute
Troy, NY 12180, USA
jih@rpi.edu

Abstract

We develop a system that lets people overcome language barriers by letting them speak a language they do not know. Our system accepts text entered by a user, translates the text, then converts the translation into a phonetic spelling in the user's own orthography. We trained the system on phonetic spellings in travel phrasebooks.

1 Introduction

Can people speak a language they don't know? Actually, it happens frequently. Travel phrasebooks contain phrases in the speaker's language (e.g., "thank you") paired with foreign-language translations (e.g., "спасибо"). Since the speaker may not be able to pronounce the foreign-language orthography, phrasebooks additionally provide phonetic spellings that approximate the sounds of the foreign phrase. These spellings employ the familiar writing system and sounds of the speaker's language. Here is a sample entry from a French phrasebook for English speakers:

English: Leave me alone.
French: Laissez-moi tranquille.
Franglish: Less-ay mwah trahn-KEEL.

The user ignores the French and goes straight to the Franglish. If the Franglish is well designed, an English speaker can pronounce it and be understood by a French listener.

Figure 1 shows a sample entry from another book—an English phrasebook for Chinese speakers. If a Chinese speaker wants to say “非常感谢你这顿美餐”，she need only read off the Chinglish “三可油否热斯弯德否米欧”，which approximates the sounds of “Thank you for this wonderful meal” using Chinese characters.

Phrasebooks permit a form of accurate, personal, oral communication that speech-to-speech

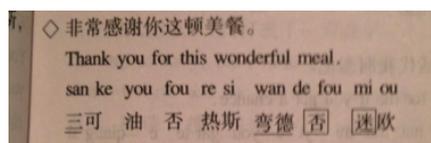


Figure 1: Snippet from phrasebook

translation devices lack. However, the user is limited to a small set of fixed phrases. In this paper, we lift this restriction by designing and evaluating a software program with the following:

- Input: Text entered by the speaker, in her own language.
- Output: Phonetic rendering of a foreign-language translation of that text, which, when pronounced by the speaker, can be understood by the listener.

The main challenge is that different languages have different orthographies, different phoneme inventories, and different phonotactic constraints, so mismatches are inevitable. Despite this, the system's output should be both unambiguously pronounceable by the speaker and readily understood by the listener.

Our goal is to build an application that covers many language pairs and directions. The current paper describes a single system that lets a Chinese person speak English.

We take a statistical modeling approach to this problem, as is done in two lines of research that are most related. The first is machine transliteration (Knight and Graehl, 1998), in which names and technical terms are translated across languages with different sound systems. The other is respelling generation (Hauer and Kondrak, 2013), where an English speaker is given a phonetic hint about how to pronounce a rare or foreign word to another English speaker. By contrast, we aim

Chinese	已经八点了
English	It's eight o'clock now
Chinglish	意思埃特额克劳克闹 (yi si ai te e ke lao ke nao)
Chinese	这件衬衫又时髦又便宜
English	this shirt is very stylish and not very expensive
Chinglish	迪思舍特意思危锐思掉利失安的闹特危锐伊克思班西五
Chinese	我们外送的最低金额是15美金
English	our minimum charge for delivery is fifteen dollars
Chinglish	奥儿米尼们差只佛低利沃锐意思发五听到乐思

Table 1: Examples of <Chinese, English, Chinglish> tuples from a phrasebook.

to help people issue full utterances that cross language barriers.

2 Evaluation

Our system's input is Chinese. The output is a string of Chinese characters that approximate English sounds, which we call Chinglish. We build several candidate Chinese-to-Chinglish systems and evaluate them as follows:

- We compute the normalized edit distance between the system's output and a human-generated Chinglish reference.
- A Chinese speaker pronounces the system's output out loud, and an English listener takes dictation. We measure the normalized edit distance against an English reference.
- We automate the previous evaluation by replace the two humans with: (1) a Chinese speech synthesizer, and (2) a English speech recognizer.

3 Data

We seek to imitate phonetic transformations found in phrasebooks, so phrasebooks themselves are a good source of training data. We obtained a collection of 1312 <Chinese, English, Chinglish> phrasebook tuples¹ (see Table 1).

We use 1182 utterances for training, 65 for development, and 65 for test. We know of no other computational work on this type of corpus.

Our Chinglish has interesting gross empirical properties. First, because Chinglish and Chinese are written with the same characters, they render the same inventory of 416 distinct syllables. However, the distribution of Chinglish syllables differs

¹Dataset can be found at <http://www.isi.edu/natural-language/mt/chinglish-data.txt>

a great deal from Chinese (Table 2). Syllables “si” and “te” are very popular, because while consonant clusters like English “st” are impossible to reproduce exactly, the particular vowels in “si” and “te” are fortunately very weak.

Frequency Rank	Chinese	Chinglish
1	de	si
2	shi	te
3	yi	de
4	ji	yi
5	zhi	fu

Table 2: Top 5 frequent syllables in Chinese (McEnery and Xiao, 2004) and Chinglish

We find that multiple occurrences of an English word type are generally associated with the same Chinglish sequence. Also, Chinglish characters do not generally span multiple English words. It is reasonable for “can I” to be rendered as “kan nai”, with “nai” spanning both English words, but this is rare.

4 Model

We model Chinese-to-Chinglish translation with a cascade of weighted finite-state transducers (wFST), shown in Figure 2. We use an online MT system to convert Chinese to an English word sequence (Eword), which is then passed through FST A to generate an English sound sequence (Epron). FST A is constructed from the CMU Pronouncing Dictionary (Weide, 2007).

Next, wFST B translates English sounds into Chinese sounds (Pinyin-split). Pinyin is an official syllable-based romanization of Mandarin Chinese characters, and Pinyin-split is a standard separation of Pinyin syllables into initial and final parts. Our wFST allows one English sound token to map

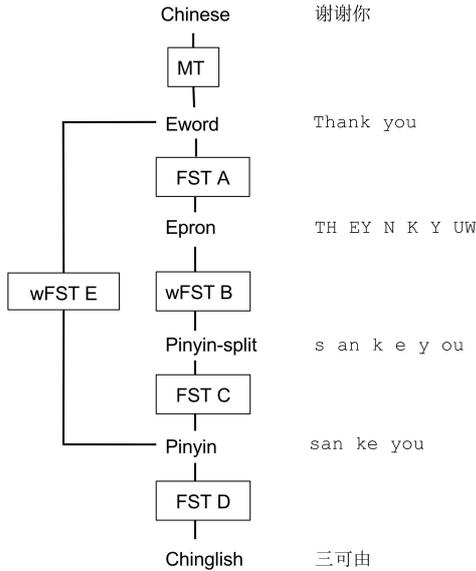


Figure 2: Finite-state cascade for modeling the relation between Chinese and Chinglish.

to one or two Pinyin-split tokens, and it also allows two English sounds to map to one Pinyin-split token.

Finally, FST C converts Pinyin-split into Pinyin, and FST D chooses Chinglish characters. We also experiment with an additional wFST E that translates English words directly into Chinglish.

5 Training

FSTs A, C, and D are unweighted, and remain so throughout this paper.

5.1 Phoneme-based model

We must now estimate the values of FST B parameters, such as $P(si|S)$. To do this, we first take our phrasebook triples and construct sample string pairs $\langle \text{Epron}, \text{Pinyin-split} \rangle$ by pronouncing the phrasebook English with FST A, and by pronouncing the phrasebook Chinglish with FSTs D and C. Then we run the EM algorithm to learn FST B parameters (Table 3) and Viterbi alignments, such as:

$$\begin{array}{c|c|c|c} g & r & ae\ n & d \\ ge & r & uan & de \end{array}$$

5.2 Phoneme-phrase-based model

Mappings between phonemes are context-sensitive. For example, when we decode English “grandmother”, we get:

labeled Epron	Pinyin-split	$P(p e)$
d	d	0.46
	d e	0.40
	d i	0.06
	s	0.01
ao r	u	0.26
	o	0.13
	ao	0.06
	ou	0.01

Table 3: Learned translation tables for the phoneme based model

$$\begin{array}{c|c|c|c|c|c|c|c} g & r & ae\ n & d & m & ah & dh & er \\ ge & r & an & de & mu & e & d & e \end{array}$$

where as the reference Pinyin-split sequence is:

$$g\ e\ r\ uan\ d\ e\ m\ a\ d\ e$$

Here, “ae n” should be decoded as “uan” when preceded by “r”. Following phrase-based methods in statistical machine translation (Koehn et al., 2003) and machine transliteration (Finch and Sumita, 2008), we model substitution of longer sequences. First, we obtain Viterbi alignments using the phoneme-based model, e.g.:

$$\begin{array}{c|c|c|c|c|c|c|c} g & r & ae\ n & d & m & ah & dh & er \\ ge & r & uan & de & m & a & d & e \end{array}$$

Second, we extract phoneme phrase pairs consistent with these alignments. We use no phrase-size limit, but we do not cross word boundaries. From the example above, we pull out phrase pairs like:

$$\begin{array}{l} g \rightarrow g\ e \\ g\ r \rightarrow g\ e\ r \\ \dots \\ r \rightarrow r \\ r\ ae\ n \rightarrow r\ uan \\ \dots \end{array}$$

We add these phrase pairs to FST B, and call this the phoneme-phrase-based model.

5.3 Word-based model

We now turn to wFST E, which short-cuts directly from English words to Pinyin. We create $\langle \text{English}, \text{Pinyin} \rangle$ training pairs from our phrasebook simply by pronouncing the Chinglish with FST D. We initially allow each English word type to map to any sequence of Pinyin, up to length 7, with uniform probability. EM learns values for parameters like $P(\text{nai te}|\text{night})$, plus Viterbi alignments such as:

Model	Top-1 Overall Average Edit Distance	Top-1 Valid Average Edit Distance	Coverage
Word based	0.664	0.042	29/65
Word-based hybrid training	0.659	0.029	29/65
Phoneme based	0.611	0.583	63/65
Phoneme-phrase based	0.194	0.136	63/65
Hybrid training and decoding	0.175	0.115	63/65

Table 4: English-to-Pinyin decoding accuracy on a test set of 65 utterances. Numbers are average edit distances between system output and Pinyin references. Valid average edit distance is calculated based only on valid outputs (e.g. 29 outputs for word based model).

accept	tips
a ke sha pu	te ti pu si

Notice that this model makes alignment errors due to sparser data (e.g., the word “tips” and “ti pu si” only appear once each in the training data).

5.4 Hybrid training

To improve the accuracy of word-based EM alignment, we use the phoneme based model to decode each English word in the training data to Pinyin. From the 100-best list of decodings, we collect combinations of start/end Pinyin syllables for the word. We then modify the initial, uniform English-to-Pinyin mapping probabilities by giving higher initial weight to mappings that respect observed start/end pairs. When we run EM, we find that alignment errors for “tips” in section 5.3 are fixed:

accept	tips
a ke sha pu te	ti pu si

5.5 Hybrid decoding

The word-based model can only decode 29 of the 65 test utterances, because wFST E fails if an utterance contains a new English word type, previously unseen in training. The phoneme-based models are more robust, able to decode 63 of the 65 utterances, failing only when some English word type falls outside the CMU pronouncing dictionary (FST A).

Our final model combines these two, using the word-based model for known English words, and the phoneme-based models for unknown English words.

6 Experiments

Our first evaluation (Table 4) is intrinsic, measuring our Chinglish output against references from

the test portion of our phrasebook, using edit distance. Here, we start with reference English and measure the accuracy of Pinyin syllable production, since the choice of Chinglish character does not affect the Chinglish pronunciation. We see that the Word-based method has very high accuracy, but low coverage. Our best system uses the Hybrid training/decoding method. As Table 6 shows, the ratio of unseen English word tokens is small, thus large portion of tokens are transformed using word-based method. The average edit distance of phoneme-phrase model and that of hybrid training/decoding model are close, indicating that long phoneme-phrase pairs can emulate word-pinyin mappings.

	Unseen	Total	Ratio
Word Type	62	249	0.249
Token	62	436	0.142

Table 6: Unseen English word type and tokens in test data.

Model	Valid Average Edit Distance
Reference English	0.477
Phoneme based	0.696
Hybrid training and decoding	0.496

Table 7: Chinglish-to-English accuracy in dictation task.

Our second evaluation is a dictation task. We speak our Chinglish character sequence output aloud and ask an English monolingual person to transcribe it. (Actually, we use a Chinese synthesizer to remove bias.) Then we measure edit distance between the human transcription and the reference English from our phrasebook. Results are shown in Table 7.

Chinese	年夜饭都要吃些什么
Reference English	what do you have for the Reunion dinner
Reference Chinglish	沃特杜又海夫佛则锐又尼恩低呢
Hybrid training/decoding Chinglish	我忒度优嗨佛佛得瑞优你恩低呢
Dictation English	what do you have for the reunion dinner
ASR English	what do you high for 43 Union Cena
Chinese	等等我
Reference English	wait for me
Reference Chinglish	唯特佛密 (wei te fo mi)
Hybrid training/decoding Chinglish	位忒佛密 (wei te fo mi)
Dictation English	wait for me
ASR English	wait for me

Table 5: Chinglish generated by hybrid training and decoding method and corresponding recognized English by dictation and automatic synthesis-recognition method.

Model	Valid Average Edit Distance
Word based	0.925
Word-based hybrid training	0.925
Phoneme based	0.937
Phoneme-phrase based	0.896
Hybrid training and decoding	0.898

Table 8: Chinglish-to-English accuracy in automatic synthesis-recognition (ASR) task. Numbers are average edit distance between recognized English and reference English.

Finally, we repeat the last experiment, but removing the human from the loop, using both automatic Chinese speech synthesis and English speech recognition. Results are shown in Table 8. Speech recognition is more fragile than human transcription, so edit distances are greater. Table 5 shows a few examples of the Chinglish generated by the hybrid training and decoding method, as well as the recognized English from the dictation and ASR tasks.

7 Conclusions

Our work aims to help people speak foreign languages they don't know, by providing native phonetic spellings that approximate the sounds of foreign phrases. We use a cascade of finite-state transducers to accomplish the task. We improve the model by adding phrases, word boundary constraints, and improved alignment.

In the future, we plan to cover more language pairs and directions. Each target language raises

interesting new challenges that come from its natural constraints on allowed phonemes, syllables, words, and orthography.

References

- Andrew Finch and Eiichiro Sumita. 2008. Phrase-based machine transliteration. In *Proceedings of the Workshop on Technologies and Corpora for Asia-Pacific Speech Translation (TCAST)*, pages 13–18.
- Bradley Hauer and Grzegorz Kondrak. 2013. Automatic generation of English respellings. In *Proceedings of NAACL-HLT*, pages 634–643.
- Kevin Knight and Jonathan Graehl. 1998. Machine transliteration. *Computational Linguistics*, 24(4):599–612.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics.
- Anthony McEnery and Zhonghua Xiao. 2004. The lancaster corpus of Mandarin Chinese: A corpus for monolingual and contrastive language study. *Religion*, 17:3–4.
- R Weide. 2007. The CMU pronunciation dictionary, release 0.7a.

Assessing the Discourse Factors that Influence the Quality of Machine Translation

Junyi Jessy Li
University of Pennsylvania
ljunyi@seas.upenn.edu

Marine Carpuat
National Research Council Canada
marine.carpuat@nrc.gc.ca

Ani Nenkova
University of Pennsylvania
nenkova@seas.upenn.edu

Abstract

We present a study of aspects of discourse structure — specifically discourse devices used to organize information in a sentence — that significantly impact the quality of machine translation. Our analysis is based on manual evaluations of translations of news from Chinese and Arabic to English. We find that there is a particularly strong mismatch in the notion of what constitutes a sentence in Chinese and English, which occurs often and is associated with significant degradation in translation quality. Also related to lower translation quality is the need to employ multiple explicit discourse connectives (*because*, *but*, etc.), as well as the presence of ambiguous discourse connectives in the English translation. Furthermore, the mismatches between discourse expressions across languages significantly impact translation quality.

1 Introduction

In this study we examine how the use of discourse devices to organize information in a sentence — and the mismatch in their usage across languages — influence machine translation (MT) quality. The goal is to identify discourse processing tasks with high potential for improving translation systems.

Historically MT researchers have focused their attention on the mismatch of linear realization of syntactic arguments (Galley et al., 2004; Collins et al., 2005), lexico-morphological mismatch (Minkov et al., 2007; Habash and Sadat, 2006) and word polysemy (Carpuat and Wu, 2007; Chan et al., 2007). Discourse structure has largely been considered irrelevant to MT, mostly due to the assumption that discourse analysis is needed to inter-

pret multi-sentential text while statistical MT systems are trained to translate a single sentence in one language into a single sentence in another.

However, discourse devices are at play in the organization of information into complex sentences. The mere definition of sentence may differ across languages. Chinese for example is anecdotally known to allow for very long sentences which at times require the use of multiple English sentences to express the same content and preserve grammaticality. Similarly discourse connectives like *because*, *but*, *since* and *while* often relate information expressed in simple sentential clauses. There are a number of possible complications in translating these connectives: they may be ambiguous between possible senses, e.g., English *while* is ambiguous between COMPARISON and TEMPORAL; explicit discourse connectives may be translated into implicit discourse relations or translated in morphology rather than lexical items (Meyer and Webber, 2013; Meyer and Poláková, 2013).

In our work, we quantify the relationship between information packaging, discourse devices, and translation quality.

2 Data and experiment settings

We examine the quality of translations to English from Chinese and Arabic using Human-targeted Translation Edit Rates (HTER) (Snover et al., 2006), which roughly captures the minimal number of edits necessary to transform the system output into an acceptable English translation of the source sentence. By comparing MT output with post-edited references, HTER provides more reliable estimates of translation quality than using translated references, especially at the segment level. The data for the analysis is drawn from an extended set of newswire reports in the 2008/2010 NIST Metrics for Machine Translation

GALE Evaluation set¹. For Chinese, there are 305 sentences (segments) translated to English by three different translation systems. For Arabic, there are 363 Arabic sentences (segments) translated by two systems.

The presence of discourse devices is analyzed only on the English side: the reference, the system hypothesis and its edited translation. Discourse connectives and their senses are identified using existing tools developed for English. Beyond its practical limitations, analyzing the reference interestingly reflects the choices made by the human translator: whether to choose to use a discourse connective, or to insert one to make an implicit relation on the source side explicit on the target side.

We first conduct analysis of variance (ANOVA) with HTER as dependent variable and the discourse factors as independent variables, and systems as subjects. We examine within-subject significance in each ANOVA model. For discourse factors that are significant at the 95% confidence level or higher according to the ANOVA analysis, we provide detailed breakdown of the system HTER for each value of the discourse factor.

In this paper we do not compare the performance of individual systems, but instead seek to understand if a discourse phenomena is problematic across systems.²

3 Sentence length and HTER

The presence of complex discourse structure is likely to be associated with longer sentences. It stands to reason that long sentences will be harder to process automatically and this reasoning has motivated the first approaches to text simplification (Chandrasekar et al., 1996). So before turning to the analysis of discourse phenomena, we examine the correlation between translation quality and sentence length. A strong correlation between the two would call for revival of interest in text simplification where syntactically complex sentences are transformed into several shorter sentences as a preprocessing step.

We find however that no strong relationship exists between the two, as shown by the correlation coefficients between HTER values and the number of words in each segment in Table 1.

¹Data used in this work includes more documents and the human edits not present in the official release.

²For the readers with keen interest in system comparison, we note that according to ANOVA none of the differences in system performance on this data is statistically significant.

Lan.	Sys1	Sys2	Sys3
ZH	0.097 (0.099)	0.117 (0.152)	0.144 (0.173)
AR	0.071(0.148)	-0.089 (-0.029)	-

Table 1: Pearson (Spearman) correlation coefficient between segment length and HTER values.

Next we examine if sentence–discourse divergence between languages and the presence of (ambiguous) discourse connectives would be more indicative of the expected translation quality.

4 When a sentence becomes discourse

Some languages allow more information to be packed into a single sentence than is possible in another language, making single-sentence translations cumbersome and often ungrammatical. Chinese is known for sentences of this kind; for example, the usage of punctuation is very different in Chinese in the sense that a comma can sometimes function as a full stop in English, motivating a series of disambiguation tasks (Jin et al., 2004; Xue and Yang, 2011; Xu and Li, 2013). Special handling of long Chinese sentences were also shown to improve machine translation (Jin and Liu, 2010; Yin et al., 2007).

To investigate the prevalence of sentences in the source language (Chinese and Arabic in our case) that do not confirm to the notion of sentence in the target language (English for the purposes of this study), we separate the translation segments in the source language into two classes: a source sentence is considered 1-1 if the reference translation consists of exactly one sentence, and 1-many if the reference contains more than one sentence.

For Chinese, 26.2% of the source segments are 1-many. These sentences tend to be much longer than average (36.6% of all words in all reference translations are part of such segments). For Arabic, the numbers are 15.2% and 26.3%, respectively. Below is an example of a 1-many Chinese segment, along with the human reference and its translation by one of the systems:

[source] 俄警方宣称，Erinys有一重要竞争对手RISC，利特维年科生前最后见面的人卢戈沃伊与友人都是从事这些行业。

[ref] Russian police claim that Erinys has an important competitor RISC. The last people Litvinenko saw while he was alive, Lugovoi and his friends, were all engaged in these industries.

[sys] Russian police have claimed that a major competitor, Litvinenko his last meeting with friends are engaged in these industries.

We conducted ANOVA on HTER, separately for each language, with type of segment (1-1 or

AOV	Arabic	Chinese
$Pr(> F)$	0.209	0.0045*

	1-1	1-many
System	HTER	HTER
ZH-Sys1	16.22	19.03*
ZH-Sys2	19.54	21.02
ZH-Sys3	20.64	23.86*

Table 2: ANOVA for both languages; average HTER for the three Chinese to English systems, stratified on type of segment (1-1 and 1-many). An (*) denotes significance at $p < 0.05$.

1-many) as the independent variable and systems treated as subjects. The test revealed that there is a significant difference in translation quality between 1-1 and 1-many segments for Chinese but not for Arabic. For the Chinese to English systems we further ran a Wilcoxon rank sum test to identify the statistical significance in performance for individual systems. For two of the three systems the difference is significant, as shown in Table 2.

We have now established that 1-many segments in Chinese to English translation are highly prevalent and their translations are of consistently lower quality compared to 1-1 segments. This finding suggests a cross language discourse analysis task of identifying Chinese sentences that cannot be translated into single English sentences. This task may be related to existing efforts in comma disambiguation in Chinese (Jin et al., 2004; Xue and Yang, 2011; Xu and Li, 2013) but the relationship between the two problems needs to be clarified in follow up work. Once 1-many segments are identified, source-side text simplification techniques may be developed (Siddharthan, 2006) to improve translation quality.

5 Explicit discourse relations

Explicit discourse relations such as COMPARISON, CONTINGENCY or TEMPORAL are signaled by an explicit connective, i.e., *however* or *because*. The Penn Discourse Treebank (PDTB) (Prasad et al., 2008) provides annotations for the arguments and relation senses of one hundred pre-selected discourse connectives over the news portion of the Penn Treebank corpus (Marcus et al., 1993). Based on the PDTB, accurate systems for explicit discourse relation identification have been developed (Pitler and Nenkova, 2009; Lin et al., 2014). The accuracy of these systems is 94% or higher, close to human performance on the task. Here we

AOV	Arabic	Chinese
$Pr(> F)$	0.39	0.0058*

		No Conn	> 1 Conn
all	% data (ZH)	53.77	15.08
1-many	% data (ZH)	13.77	5.25
		HTER mean	HTER mean
all	ZH-Sys1	16.11	19.84 ⁺
	ZH-Sys2	19.96	22.39
	ZH-Sys3	20.70	25.00*
1-many	ZH-Sys1	16.94	22.75 ⁺
	ZH-Sys2	20.47	23.25
	ZH-Sys3	22.30	29.68*

Table 3: Number of connectives: ANOVA for both languages; proportion of data in each factor level and average HTER for the three Chinese-English systems, of the entire dataset and of 1-many translations. An (*) or (+) sign denotes significance at 95% and 90% confidence levels, respectively.

study the influence of explicit discourse relations on machine translation quality and their interaction with 1-1 and 1-many segments.

5.1 Number of connectives

We identify discourse connectives and their senses (TEMPORAL, COMPARISON, CONTINGENCY or EXPANSION) in each reference segment using the system in Pitler and Nenkova (2009)³. We compare the translation quality obtained on segments with reference translation containing no discourse connective, exactly one discourse connective and more than one discourse connective.

The ANOVA indicates that the number of connectives is not a significant factor for Arabic translation, but significantly impacts Chinese translation quality. A closer inspection using Wilcoxon rank sum tests reveals that the difference in translation quality is statistically significant only between the groups of segments with no connective vs. those with more than one connective. Additionally, we ran Wilcoxon rank sum test over 1-1 and 1-many segments individually and find that the presence of discourse connectives is associated with worse quality only in the latter case. Effects above are illustrated in Table 3.

5.2 Ambiguity of connectives

A number of discourse connectives are ambiguous with respect to the discourse relation they convey. For example, *while* can signal either COMPARI-

³<http://www.cis.upenn.edu/~epitler/discourse.html>; We used the Stanford Parser (Klein and Manning, 2003).

AOV	Arabic	Chinese
$Pr(> F)$	0.57	0.00014*

	has-amb-conn	no-amb-conn
System	HTER mean	HTER mean
ZH-Sys1	21.57	16.34*
ZH-Sys2	21.44	19.72
ZH-Sys3	27.47	20.69*

Table 4: ANOVA for both languages; average HTER for the three Chinese systems for segments with (11.80% of all data) and without an ambiguous connective in the reference translation. An (*) denotes significance at $p < 0.05$.

SON or TEMPORAL relations and *since* can signal either CONTINGENCY or TEMPORAL. In translation this becomes a problem when the ambiguity is present in one language but not in the other. In such cases the sense in source ought to be disambiguated before translation. Here we compare the translation quality of segments which contain ambiguous discourse connectives in the reference translation to those that do not. This analysis gives lower bound on the translation quality degradation associated with discourse phenomena as it does not capture problems arising from connective ambiguity on the source side.

We base our classification of discourse connectives into ambiguous or not according to the distribution of their senses in the PDTB. We call a connective ambiguous if its most frequent sense among COMPARISON, CONTINGENCY, EXPANSION, TEMPORAL accounts for less than 80% of occurrence of that connective in the PDTB. Nineteen connectives meet this criterion of ambiguity.⁴

In the ANOVA tests for each language, we compared the quality of segments which contained an ambiguous connective in the reference with those that do not, with systems treated as subjects. For Arabic the presence of ambiguous connective did not yield a statistically significant difference. The difference however was highly significant for Chinese, as shown in Table 4.

The finding that discourse connective ambiguity is associated with change in translation quality for Chinese but not for Arabic is rather interesting. It appears that the language pair in translation impacts the expected gains from discourse analysis on translation.

⁴The ambiguous connectives are: as, as if, as long as, as though, finally, if and when, in the end, in turn, lest, meanwhile, much as, neither...nor, now that, rather, since, ultimately, when, when and if, while

AOV	Event	Arabic	Chinese
$Pr(> F)$	Contingency Comp.:Temp.	0.61 0.047*	0.028* 0.0041*

Chinese	HTER	HTER
	Contingency	\neg Contingency
Sys1	20.15	16.72
Sys2	21.69	19.80
Sys3	25.87	21.16 ⁺
	Comp.^Temp.	\neg (Comp.^Temp.)
Sys1	23.58	16.64*
Sys2	26.16	19.63*
Sys3	27.20	21.21 ⁺

Table 5: ANOVA for both languages; average HTER for Chinese sentences containing a CONTINGENCY relation (6.89% of all data) or both COMPARISON and TEMPORAL (4.59% of all data). An (*) or (+) sign denotes significance at 95% and 90% confidence levels, respectively.

5.3 Relation senses

Here we study whether discourse relations of specific senses pose more difficulties on translations than others and whether there are interactions between senses. In the ANOVA analysis we used a binary factor for each of the four possible senses. For example, we compare the translation quality of segments that contain COMPARISON relations in the reference translation with those that do not.

The relation sense makes a significant difference in translation quality for Chinese but not for Arabic. For Chinese specifically sentences that express CONTINGENCY relations have worse quality translations than sentences that do not express CONTINGENCY. One explanation for this tendency may be that CONTINGENCY in Chinese contains more ambiguity with other relations such as TEMPORAL, as tense is expressed lexically in Chinese (no morphological tense marking on verbs). Finally, the interaction between COMPARISON and TEMPORAL is significant for both languages.

Table 5 shows the effect of relation sense on HTER values for Chinese.

6 Human edits of discourse connectives

A relation expressed implicitly without a connective in one language may need to be explicit in another. Moreover, the expressions themselves are used differently; for example, the paired connective “虽然...但是” (despite...but) in Chinese should not be translated into two redundant connectives in English. It is also possible that the source language contains an explicit discourse

connective which is not translated in the target language, as has been quantitatively studied recently by Meyer and Webber (2013). An example from our dataset is shown below:

[source] 还有些人可到大学的游戏专业深造，而后被聘请为大游戏厂商的技术顾问等。

[ref] Still some others can receive further professional game training in universities and later(*Temporal*) be employed as technical consultants by large game manufacturers, etc.

[sys] Some people may go to the university games professional education, which is appointed as the big game manufacturers such as technical advisers.

[edited] Some people may go to university to receive professional game education, and later(*Temporal*) be appointed by the big game manufacturers as technical advisers.

The system fails to translate the discourse connective “而后” (later), leading to a probable misinterpretation between receiving education and being appointed as technical advisers.

Due to the lack of reliable tools and resources, we approximate mismatches between discourse expressions in the source and MT output using discourse-related edits. We identify explicit discourse connectives and their senses in the system translation and the human edited version of that translation. Then we consider the following mutually exclusive possibilities: (i) there are no discourse connectives in either the system output or the edit; (ii) the system output and its edited version contain exactly the same discourse connectives with the same senses; (iii) there is a discourse connective present in the system output but not in the edit or vice versa. In the ANOVA we use a factor with three levels corresponding to the three cases described above. The factor is significant for both Chinese and Arabic. In both languages, the mismatch case (iii) involves significantly higher HTER than either case (i) or (ii). The human edit rate in the mismatch class is on average four points greater than that in the other classes.

Obviously, the mismatch in implicit/explicit expression of discourse relation is related to the first problem we studied, i.e., if the source segment is translated into one or multiple sentences in English, since discourse relations between adjacent sentences are more often implicit (than intra-sentence ones). For this reason we performed a Wilcoxon rank sum test for the translation quality of segments with discourse mismatch conditioned on whether the segment was 1-1 or 1-many. For both languages a significant difference was found for 1-1 sentences but not 1-many. Table 6 shows the proportion of data in each of the conditioned classes and the average HTER for sen-

% data	Mismatch	Mismatch (1-1)	¬Mismatch (1-1)
Arabic	21.27	15.47	69.34
Chinese	29.51	17.05	56.82

AOV	Arabic	Chinese
$Pr(> F)$	$4.0 \times 10^{-6*}$	$4.1 \times 10^{-11*}$

	HTER	
	¬ Mismatch	Mismatch
AR-Sys1	11.23	15.92*
AR-Sys2	11.64	15.74*
ZH-Sys1	15.57	20.72*
ZH-Sys2	19.02	22.34*
ZH-Sys3	11.64	15.74*
	¬ Mismatch 1-1	Mismatch 1-1
AR-Sys1	10.86	16.24*
AR-Sys2	11.58	16.65*
ZH-Sys1	15.47	19.13*
ZH-Sys2	18.68	22.52*
ZH-Sys3	19.57	26.07*

Table 6: Data portions, ANOVA for both languages and average HTER for segments where there is a discourse mismatch between system and edited translations. An (*) denotes significance at $p < 0.05$.

tences from the mismatch case (iii) where a discourse connective was edited and the others (no such edits). Translation quality degrades significantly for all systems for the mismatch case, over all data as well as 1-1 segments.

7 Conclusion

We showed that translation from Chinese to English is made more difficult by various discourse events such as the use of discourse connectives, the ambiguity of the connectives and the type of relations they signal. None of these discourse factors has a significant impact on translation quality from Arabic to English. Translation quality from both languages is adversely affected by translations of discourse relations expressed implicitly in one language but explicitly in the other or by paired connectives. Our experiments indicate that discourse usage may affect machine translation between some language pairs but not others, and for particular relations such as CONTINGENCY. Finally, we established the need to identify sentences in the source language that would be translated into multiple sentences in English. Especially in translating from Chinese to English, there is a large number of such sentences which are currently translated much worse than other sentences.

References

- Marine Carpuat and Dekai Wu. 2007. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72.
- Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 33–40.
- Raman Chandrasekar, Christine Doran, and Bangalore Srinivas. 1996. Motivations and methods for text simplification. In *Proceedings of the 16th Conference on Computational Linguistics (COLING)*, pages 1041–1044.
- Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 531–540.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What’s in a translation rule? In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 273–280.
- Nizar Habash and Fatiha Sadat. 2006. Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Short Papers*, pages 49–52.
- Yaohong Jin and Zhiying Liu. 2010. Improving Chinese-English patent machine translation using sentence segmentation. In *International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE)*, pages 1–6.
- Meixun Jin, Mi-Young Kim, Dongil Kim, and Jong-Hyeok Lee. 2004. Segmentation of Chinese long sentences using commas. In *Proceedings of the Third SIGHAN Workshop on Chinese Language Processing (SIGHAN)*, pages 1–8.
- Dan Klein and Christopher D. Manning. 2003. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems*, volume 15.
- Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering*, 20:151–184, 4.
- Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics - Special issue on using large corpora*, 19(2):313–330.
- Thomas Meyer and Lucie Poláková. 2013. Machine translation with many manually labeled discourse connectives. In *Proceedings of the Workshop on Discourse in Machine Translation (DiscoMT)*, pages 43–50.
- Thomas Meyer and Bonnie Webber. 2013. Implication of discourse connectives in (machine) translation. In *Proceedings of the Workshop on Discourse in Machine Translation (DiscoMT)*, pages 19–26.
- Einat Minkov, Kristina Toutanova, and Hisami Suzuki. 2007. Generating complex morphology for machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 128–135.
- Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference: Short Papers*, pages 13–16.
- Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltasakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC)*.
- Advait Siddharthan. 2006. Syntactic simplification and text cohesion. *Research on Language and Computation*, 4(1):77–109.
- 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*, pages 223–231.
- Shengqin Xu and Peifeng Li. 2013. Recognizing Chinese elementary discourse unit on comma. In *International Conference on Asian Language Processing (IALP)*, pages 3–6.
- Nianwen Xue and Yaqin Yang. 2011. Chinese sentence segmentation as comma classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT): Short Papers*, pages 631–635.
- Dapeng Yin, F. Ren, Peilin Jiang, and S. Kuroiwa. 2007. Chinese complex long sentences processing method for Chinese-Japanese machine translation. In *International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE)*, pages 170–175.

Automatic Detection of Machine Translated Text and Translation Quality Estimation

Roe Aharoni

Dept. of Computer Science
Bar Ilan University
Ramat-Gan, Israel 52900
roee.aharoni@gmail.com

Moshe Koppel

Dept. of Computer Science
Bar Ilan University
Ramat-Gan, Israel 52900
moishk@gmail.com

Yoav Goldberg

Dept. of Computer Science
Bar Ilan University
Ramat-Gan, Israel 52900
yoav.goldberg@gmail.com

Abstract

We show that it is possible to automatically detect machine translated text at sentence level from monolingual corpora, using text classification methods. We show further that the accuracy with which a learned classifier can detect text as machine translated is strongly correlated with the translation quality of the machine translation system that generated it. Finally, we offer a generic machine translation quality estimation technique based on this approach, which does not require reference sentences.

1 Introduction

The recent success and proliferation of statistical machine translation (MT) systems raise a number of important questions. Prominent among these are how to evaluate the quality of such a system efficiently and how to detect the output of such systems (for example, to avoid using it circularly as input for refining MT systems).

In this paper, we will answer both these questions. First, we will show that using style-related linguistic features, such as frequencies of parts-of-speech n-grams and function words, it is possible to learn classifiers that distinguish machine-translated text from human-translated or native English text. While this is a straightforward and not entirely novel result, our main contribution is to relativize the result. We will see that the success of such classifiers are strongly correlated with the quality of the underlying machine translation system. Specifically, given a corpus consisting of both machine-translated English text (English being the target language) and native English text (not necessarily the reference translation of the machine-translated text), we measure the accuracy of the system in classifying the sentences in the

corpus as machine-translated or not. This accuracy will be shown to decrease as the quality of the underlying MT system increases. In fact, the correlation is strong enough that we propose that this accuracy measure itself can be used as a measure of MT system quality, obviating the need for a reference corpus, as for example is necessary for BLEU (Papineni et al., 2001).

The paper is structured as follows: In the next section, we review previous related work. In the third section, we describe experiments regarding the detection of machine translation and in the fourth section we discuss the use of detection techniques as a machine translation quality estimation method. In the final section we offer conclusions and suggestions for future work.

2 Previous Work

2.1 Translationese

The special features of translated texts have been studied widely for many years. Attempts to define their characteristics, often called "Translation Universals", include (Toury, 1980; Blum-Kulka and Levenston, 1983; Baker, 1993; Gellerstam, 1986). The differences between native and translated texts found there go well beyond systematic translation errors and point to a distinct "Translationese" dialect.

Using automatic text classification methods in the field of translation studies had many use cases in recent years, mainly as an empirical method of measuring, proving or contradicting translation universals. Several works (Baroni and Bernardini, 2006; Kurokawa et al., 2009; Ilisei et al., 2010) used text classification techniques in order to distinguish human translated text from native language text at document or paragraph level, using features like word and POS n-grams, proportion of grammatical words in the text, nouns, finite verbs, auxiliary verbs, adjectives, adverbs, nu-

merals, pronouns, prepositions, determiners, conjunctions etc. Koppel and Ordan (2011) classified texts to original or translated, using a list of 300 function words taken from LIWC (Pennebaker et al., 2001) as features. Volanski et al. (2013) also tested various hypotheses regarding "Translationese", using 32 different linguistically-informed features, to assess the degree to which different sets of features can distinguish between translated and original texts.

2.2 Machine Translation Detection

Regarding the detection of machine translated text, Carter and Inkpen (2012) translated the Hansards of the 36th Parliament of Canada using the Microsoft Bing MT web service, and conducted three detection experiments at document level, using unigrams, average token length, and type-token ratio as features. Arase and Zhou (2013) trained a sentence-level classifier to distinguish machine translated text from human generated text on English and Japanese web-page corpora, translated by Google Translate, Bing and an in-house SMT system. They achieved very high detection accuracy using application-specific feature sets for this purpose, including indicators of the "Phrase Salad" (Lopez, 2008) phenomenon or "Gappy-Phrases" (Bansal et al., 2011).

While Arase and Zhou (2013) considered MT detection at sentence level, as we do in this paper, they did not study the correlation between the translation quality of the machine translated text and the ability to detect it. We show below that such detection is possible with very high accuracy only on low-quality translations. We examine this detection accuracy vs. quality correlation, with various MT systems, such as rule-based and statistical MT, both commercial and in-house, using various feature sets.

3 Detection Experiments

3.1 Features

We wish to distinguish machine translated English sentences from either human-translated sentences or native English sentences. Due to the sparseness of the data at the sentence level, we use common content-independent linguistic features for the classification task. Our features are binary, denoting the presence or absence of each of a set of part-of-speech n-grams acquired using the Stanford POS tagger (Toutanova et al., 2003),

as well as the presence or absence of each of 467 function words taken from LIWC (Pennebaker et al., 2001). We consider only those entries that appear at least ten times in the entire corpus, in order to reduce sparsity in the data. As our learning algorithm we use SVM with sequential minimal optimization (SMO), taken from the WEKA machine learning toolkit (Hall et al., 2009).

3.2 Detecting Different MT Systems

In the first experiment set, we explore the ability to detect outputs of machine translated text from different MT systems, in an environment containing both human generated and machine translated text. For this task, we use a portion of the Canadian Hansard corpus (Germann, 2001), containing 48,914 parallel sentences from French to English. We translate the French portion of the corpus using several MT systems, respectively: Google Translate, Systran, and five other commercial MT systems available at the <http://itranslate4.eu> website, which enables to query example MT systems built by several european MT companies. After translating the sentences, we take 20,000 sentences from each engine output and conduct the detection experiment by labeling those sentences as MT sentences, and another 20,000 sentences, which are the human reference translations, labeled as reference sentences. We conduct a 10-fold cross-validation experiment on the entire 40,000 sentence corpus. We also conduct the same experiment using 20,000 random, non-reference sentences from the same corpus, instead of the reference sentences. Using simple linear regression, we also obtain an R^2 value (coefficient of determination) over the measurements of detection accuracy and BLEU score, for each of three feature set combinations (function words, POS tags and mixed) and the two data combinations (MT vs. reference and MT vs. non reference sentences). The detection and R^2 results are shown in Table 1.

As can be seen, best detection results are obtained using the full combined feature set. It can also be seen that, as might be expected, it is easier to distinguish machine-translated sentences from a non-reference set than from the reference set. In Figure 1, we show the relationship of the observed detection accuracy for each system with the BLEU score of that system. As is evident, regardless of the feature set or non-MT sentences used, the correlation between detection accuracy and BLEU

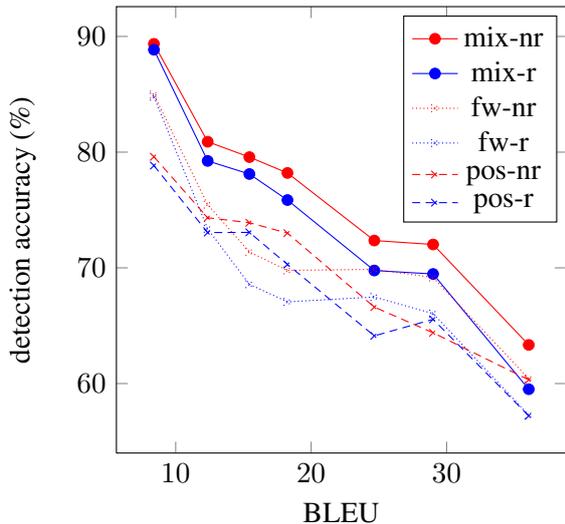


Figure 1: Correlation between detection accuracy and BLEU score on commercial MT systems, using POS, function words and mixed features against reference and non-reference sentences.

score is very high, as we can also see from the R^2 values in Table 1.

3.3 In-House SMT Systems

	Parallel	Monolingual	BLEU
SMT-1	2000k	2000k	28.54
SMT-2	1000k	1000k	27.76
SMT-3	500k	500k	29.18
SMT-4	100k	100k	23.83
SMT-5	50k	50k	24.34
SMT-6	25k	25k	22.46
SMT-7	10k	10k	20.72

Table 3: Details for Moses based SMT systems

In the second experiment set, we test our detection method on SMT systems we created, in which we have control over the training data and the expected overall relative translation quality. In order to do so, we use the Moses statistical machine translation toolkit (Koehn et al., 2007). To train the systems, we take a portion of the Europarl corpus (Koehn, 2005), creating 7 different SMT systems, each using a different amount of training data, for both the translation model and language model. We do this in order to create different quality translation systems, details of which are described in Table 3. For purposes of classification, we use the same content independent features as in the previous experiment, based on func-

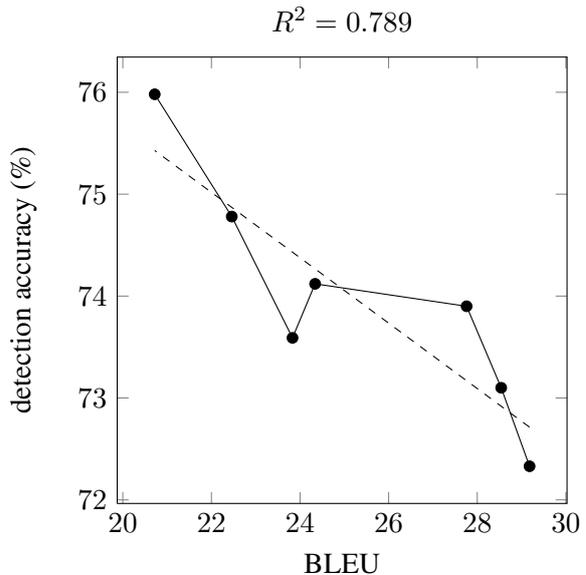


Figure 2: Correlation between detection accuracy and BLEU score on in-house Moses-based SMT systems against non-reference sentences using content independent features.

tion words and POS tags, again with SMO-based SVM as the classifier. For data, we use 20,000 random, non reference sentences from the Hansard corpus, against 20,000 sentences from one MT system per experiment, again resulting in 40,000 sentence instances per experiment. The relationship between the detection results for each MT system and the BLEU score for that system, resulting in $R^2 = 0.774$, is shown in Figure 2.

4 Machine Translation Evaluation

4.1 Human Evaluation Experiments

As can be seen in the above experiments, there is a strong correlation between the BLEU score and the MT detection accuracy of our method. In fact, results are linearly and negatively correlated with BLEU, as can be seen both on commercial systems and our in-house SMT systems. We also wish to consider the relationship between detection accuracy and a human quality estimation score. To do this, we use the French-English data from the 8th Workshop on Statistical Machine Translation - WMT13' (Bojar et al., 2013), containing outputs from 13 different MT systems and their human evaluations. We conduct the same classification experiment as above, with features based on function words and POS tags, and SMO-based SVM as the classifier. We first use 3000 refer-

Features	Data	Google	Moses	Systran	ProMT	Linguec	Skycode	Trident	R^2
mixed	MT/non-ref	63.34	72.02	72.36	78.2	79.57	80.9	89.36	0.946
mixed	MT/ref	59.51	69.47	69.77	75.86	78.11	79.24	88.85	0.944
func. w.	MT/non-ref	60.43	69.17	69.87	69.78	71.38	75.46	84.97	0.798
func. w.	MT/ref	57.27	66.05	67.48	67.06	68.58	73.37	84.79	0.779
POS	MT/non-ref	60.32	64.39	66.61	73	73.9	74.33	79.6	0.978
POS	MT/ref	57.21	65.55	64.12	70.29	73.06	73.04	78.84	0.948

Table 1: Classifier performance, including the R^2 coefficient describing the correlation with BLEU.

MT Engine	Example
Google Translate	"These days, all but one were subject to a vote, and all had a direct link to the post September 11th."
Moses	"these days , except one were the subject of a vote , and all had a direct link with the after 11 September ."
Systran	"From these days, all except one were the object of a vote, and all were connected a direct link with after September 11th."
Linguec	"Of these days, all except one were making the object of a vote and all had a straightforward tie with after September 11."
ProMT	"These days, very safe one all made object a vote, and had a direct link with after September 11th."
Trident	"From these all days, except one operated object voting, and all had a direct rope with after 11 septembre."
Skycode	"In these days, all safe one made the object in a vote and all had a direct connection with him after 11 of September."

Table 2: Outputs from several MT systems for the same source sentence (function words marked in bold)

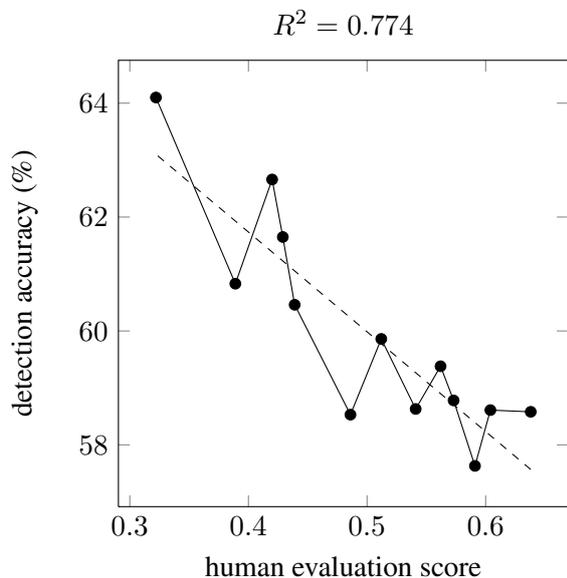


Figure 3: Correlation between detection accuracy and human evaluation scores on systems from WMT13' against reference sentences.

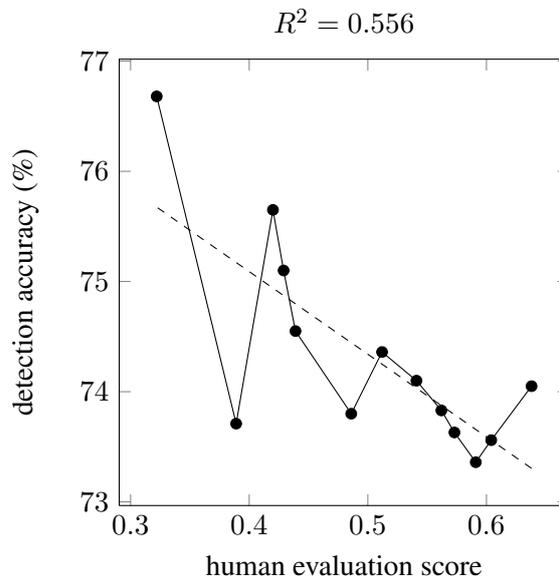


Figure 4: Correlation between detection accuracy and human evaluation scores on systems from WMT 13' against non-reference sentences.

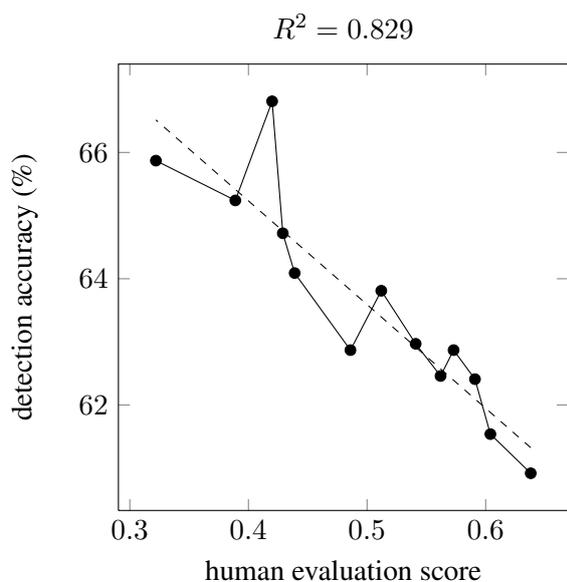


Figure 5: Correlation between detection accuracy and human evaluation scores on systems from WMT 13’ against non-reference sentences, using the syntactic CFG features described in section 4.2

ence sentences from the WMT13’ English reference translations, against the matching 3000 output sentences from one MT system at a time, resulting in 6000 sentence instances per experiment. As can be seen in Figure 3, the detection accuracy is strongly correlated with the evaluations scores, yielding $R^2 = 0.774$. To provide another measure of correlation, we compared every pair of data points in the experiment to get the proportion of pairs ordered identically by the human evaluators and our method, with a result of 0.846 (66 of 78). In the second experiment, we use 3000 random, non reference sentences from the newest 2011-2012 corpora published in WMT12’ (Callison-Burch et al., 2012) against 3000 output sentences from one MT system at a time, again resulting in 6000 sentence instances per experiment. While applying the same classification method as with the reference sentences, the detection accuracy rises, while the correlation with the translation quality yields $R^2 = 0.556$, as can be seen in Figure 4. Here, the proportion of identically ordered pairs is 0.782 (61 of 78).

4.2 Syntactic Features

We note that the second leftmost point in Figures 3, 4 is an outlier: that is, our method has a hard time detecting sentences produced by this system although it is not highly rated by human evalu-

ators. This point represents the Joshua (Post et al., 2013) SMT system. This system is syntax-based, which apparently confound our POS and FW-based classifier, despite it’s low human evaluation score. We hypothesize that the use of syntax-based features might improve results. To verify this intuition, we create parse trees using the Berkeley parser (Petrov and Klein, 2007) and extract the one-level CFG rules as features. Again, we represent each sentence as a boolean vector, in which each entry represents the presence or absence of the CFG rule in the parse-tree of the sentence. Using these features alone, without the FW and POS tag based features presented above, we obtain an $R^2 = 0.829$ with a proportion of identically ordered pairs at 0.923 (72 of 78), as shown in Figure 5.

5 Discussion and Future Work

We have shown that it is possible to detect machine translation from monolingual corpora containing both machine translated text and human generated text, at sentence level. There is a strong correlation between the detection accuracy that can be obtained and the BLEU score or the human evaluation score of the machine translation itself. This correlation holds whether or not a reference set is used. This suggests that our method might be used as an unsupervised quality estimation method when no reference sentences are available, such as for resource-poor source languages. Further work might include applying our methods to other language pairs and domains, acquiring word-level quality estimation or integrating our method in a machine translation system. Furthermore, additional features and feature selection techniques can be applied, both for improving detection accuracy and for strengthening the correlation with human quality estimation.

Acknowledgments

We would like to thank Noam Ordan and Shuly Wintner for their help and feedback on the early stages of this work. This research was funded in part by the Intel Collaborative Research Institute for Computational Intelligence.

References

- Yuki Arase and Ming Zhou. 2013. Machine translation detection from monolingual web-text. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1597–1607, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Mona Baker. 1993. Corpus linguistics and translation studies: Implications and applications. *Text and technology: in honour of John Sinclair*, 233:250.
- Mohit Bansal, Chris Quirk, and Robert C. Moore. 2011. Gappy phrasal alignment by agreement. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *ACL*, pages 1308–1317. The Association for Computer Linguistics.
- Marco Baroni and Silvia Bernardini. 2006. A new approach to the study of translationese: Machine-learning the difference between original and translated text. *LLC*, 21(3):259–274.
- Shoshana Blum-Kulka and Eddie A. Levenston. 1983. Universals of lexical simplification. *Strategies in Interlanguage Communication*, pages 119–139.
- 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, August. Association for Computational Linguistics.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada, June. Association for Computational Linguistics.
- Dave Carter and Diana Inkpen. 2012. Searching for poor quality machine translated text: Learning the difference between human writing and machine translations. In Leila Kosseim and Diana Inkpen, editors, *Canadian Conference on AI*, volume 7310 of *Lecture Notes in Computer Science*, pages 49–60. Springer.
- Martin Gellerstam. 1986. Translationese in swedish novels translated from english. In Lars Wollin and Hans Lindquist, editors, *Translation Studies in Scandinavia*, pages 88–95.
- Ulrich Germann. 2001. Aligned hansards of the 36th parliament of canada release 2001-1a.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18.
- Iustina Ilisei, Diana Inkpen, Gloria Corpas Pastor, and Ruslan Mitkov. 2010. Identification of translationese: A machine learning approach. In Alexander F. Gelbukh, editor, *CICLing*, volume 6008 of *Lecture Notes in Computer Science*, pages 503–511. Springer.
- 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 *ACL*. The Association for Computer Linguistics.
- Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand. AAMT, AAMT.
- Moshe Koppel and Noam Ordan. 2011. Translationese and its dialects. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *ACL*, pages 1318–1326. The Association for Computer Linguistics.
- David Kurokawa, Cyril Goutte, and Pierre Isabelle. 2009. Automatic Detection of Translated Text and its Impact on Machine Translation. In *Conference Proceedings: the twelfth Machine Translation Summit*.
- Adam Lopez. 2008. Statistical machine translation. *ACM Computing Surveys (CSUR)*, 40(3):8.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. Technical report, IBM Research Report.
- J.W. Pennebaker, M.E. Francis, and R.J. Booth. 2001. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*.
- Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics, Proceedings of the Main Conference*, pages 404–411.
- Marius Popescu. 2011. Studying translationese at the character level. In Galia Angelova, Kalina Bontcheva, Ruslan Mitkov, and Nicolas Nicolov, editors, *RANLP*, pages 634–639. RANLP 2011 Organising Committee.
- Matt Post, Juri Ganitkevitch, Luke Orland, Jonathan Weese, Yuan Cao, and Chris Callison-Burch. 2013. Joshua 5.0: Sparser, better, faster, server. In *Proceedings of the Eighth Workshop on Statistical Machine Translation, August 8-9, 2013.*, pages 206–212. Association for Computational Linguistics.
- Gideon Toury. 1980. *In Search of a Theory of Translation*.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *IN PROCEEDINGS OF HLT-NAACL*, pages 252–259.

Hans van Halteren. 2008. Source language markers in europarl translations. In Donia Scott and Hans Uszkoreit, editors, *COLING*, pages 937–944.

Vered Volansky, Noam Ordan, and Shuly Wintner. 2013. On the features of translationese. *Literary and Linguistic Computing*.

Improving sparse word similarity models with asymmetric measures

Jean Mark Gawron

San Diego State University
gawron@mail.sdsu.edu

Abstract

We show that asymmetric models based on Tversky (1977) improve correlations with human similarity judgments and nearest neighbor discovery for both frequent and middle-rank words. In accord with Tversky’s discovery that asymmetric similarity judgments arise when comparing sparse and rich representations, improvement on our two tasks can be traced to heavily weighting the feature bias toward the rarer word when comparing high- and mid-frequency words.

1 Introduction

A key assumption of most models of similarity is that a similarity relation is symmetric. This assumption is foundational for some conceptions, such as the idea of a similarity space, in which similarity is the inverse of distance; and it is deeply embedded into many of the algorithms that build on a similarity relation among objects, such as clustering algorithms. The symmetry assumption is not, however, universal, and it is not essential to all applications of similarity, especially when it comes to modeling human similarity judgments. Citing a number of empirical studies, Tversky (1977) calls symmetry directly into question, and proposes two general models that abandon symmetry. The one most directly related to a large body of word similarity work that followed is what he calls the **ratio model**, which defines $\text{sim}(a, b)$ as:

$$\frac{f(A \cap B)}{f(A \cap B) + \alpha f(A \setminus B) + \beta f(B \setminus A)} \quad (1)$$

Here A and B represent feature sets for the objects a and b respectively; the term in the numerator is a function of the set of shared features, a measure of

similarity, and the last two terms in the denominator measure dissimilarity: α and β are real-number weights; when $\alpha \neq \beta$, symmetry is abandoned.

To motivate such a measure, Tversky presents experimental data with asymmetric similarity results, including similarity comparisons of countries, line drawings of faces, and letters. Tversky shows that many similarity judgment tasks have an inherent asymmetry; but he also argues, following Rosch (1975), that certain kinds of stimuli are more naturally used as foci or standards than others. Goldstone (in press) summarizes the results succinctly: “Asymmetrical similarity occurs when an object with many features is judged as less similar to a sparser object than vice versa; for example, North Korea is judged to be more like China than China is [like] North Korea.” Thus, one source of asymmetry is the comparison of sparse and dense representations.

The relevance of such considerations to word similarity becomes clear when we consider that for many applications, word similarity measures need to be well-defined when comparing very frequent words with infrequent words. To make this concrete, let us consider a word representation in the word-as-vector paradigm (Lee, 1997; Lin, 1998), using a dependency-based model. Suppose we want to measure the semantic similarity of *boat*, rank 682 among the nouns in the BNC corpus studied below, which has 1057 nonzero dependency features based on 50 million words of data, with *dinghy*, rank 6200, which has only 113 nonzero features. At the level of the vector representations we are using, these are events of very different dimensionality; that is, there are ten times as many features in the representation of *boat* as there are in the representation of *dinghy*. If in Tversky/Rosch terms, the more frequent word is also a more likely focus, then this is exactly the kind of situation in which asymmetric similarity judgments will arise. Below we show that an

asymmetric measure, using α and β biased in favor of the less frequent word, greatly improves the performance of a dependency-based vector model in capturing human similarity judgments.

Before presenting these results, it will be helpful to slightly reformulate and slightly generalize Tversky's ratio model. The reformulation will allow us to directly draw the connection between the ratio model and a set of similarity measures that have played key roles in the similarity literature. First, since Tversky has primarily additive f in mind, we can reformulate $f(A \cap B)$ as follows

$$f(A \cap B) = \sum_{f \in A \cap B} \text{wght}(f) \quad (2)$$

Next, since we are interested in generalizing from sets of features, to real-valued vectors of features, w_1, w_2 , we define

$$\sigma_{\text{SI}}(w_1, w_2) = \sum_{f \in w_1 \cap w_2} \text{SI}(w_1[f], w_2[f]). \quad (3)$$

Here SI is some numerical operation on real-number feature values (SI stands for **shared information**). If the operation is MIN and $w_1[f]$ and $w_2[f]$ both contain the feature weights for f , then

$$\sum_{f \in A \cap B} \text{wght}(f) = \sigma_{\text{MIN}}(w_1, w_2) = \sum_{f \in w_1 \cap w_2} \text{MIN}(w_1[f], w_2[f]),$$

so with SI set to MIN, Equation (3) includes Equation (2) as a special case. Similarly, $\sigma(w_1, w_1)$ represents the summed feature weights of w_1 , and therefore,

$$f(w_1 \setminus w_2) = \sigma(w_1, w_1) - \sigma(w_1, w_2)$$

In this generalized form, then, (1) becomes

$$\frac{\sigma(w_1, w_2)}{\sigma(w_1, w_2) + \alpha[\sigma(w_1, w_1) - \sigma(w_1, w_2)] + \beta[\sigma(w_2, w_2) - \sigma(w_1, w_2)]} = \frac{\sigma(w_1, w_2)}{\alpha\sigma(w_1, w_1) + \beta\sigma(w_2, w_2) + \sigma(w_1, w_2) - (\alpha + \beta)\sigma(w_1, w_2)} \quad (4)$$

Thus, if $\alpha + \beta = 1$, Tversky's ratio model becomes simply:

$$\text{sim}(w_1, w_2) = \frac{\sigma(w_1, w_2)}{\alpha\sigma(w_1, w_1) + (1 - \alpha)\sigma(w_2, w_2)} \quad (5)$$

The computational advantage of this reformulation is that the core similarity operation $\sigma(w_1, w_2)$ is done on what is generally only a small number of shared features, and the $\sigma(w_i, w_i)$ calculations (which we will call self-similarities), can be computed in advance. Note that $\text{sim}(w_1, w_2)$ is symmetric if and only if $\alpha = 0.5$. When $\alpha > 0.5$,

$\text{sim}(w_1, w_2)$ is biased in favor of w_1 as the referent; When $\alpha < 0.5$, $\text{sim}(w_1, w_2)$ is biased in favor of w_2 .

Consider four similarity functions that have played important roles in the literature on similarity:

$$\begin{aligned} \text{DICE PROD}(w_1, w_2) &= \frac{2 * w_1 \cdot w_2}{\|w_1\|^2 + \|w_2\|^2} \\ \text{DICE}^\dagger(w_1, w_2) &= \frac{2 * \sum_{f \in w_1 \cap w_2} \min(w_1[f], w_2[f])}{\sum w_1[f] + \sum w_2[f]} \\ \text{LIN}(w_1, w_2) &= \frac{\sum_{f \in w_1 \cap w_2} w_1[f] + w_2[f]}{\sum w_1[f] + \sum w_2[f]} \\ \text{COS}(w_1, w_2) &= \text{DICE PROD applied to unit vectors} \end{aligned} \quad (6)$$

The function DICE PROD is not well known in the word similarity literature, but in the data mining literature it is often just called Dice coefficient, because it generalized the set comparison function of Dice (1945). Observe that cosine is a special case of DICE PROD. DICE[†] was introduced in Curran (2004) and was the most successful function in his evaluation. Since LIN was introduced in Lin (1998); several different functions have born that name. The version used here is the one used in Curran (2004).

The three distinct functions in Equation 6 have a similar form. In fact, all can be defined in terms of σ functions differing only in their SI operation.

Let σ_{SI} be a shared feature sum for operation SI, as defined in Equation (3). We define the Tversky-normalized version of σ_{SI} , written T_{SI} , as:¹

$$T_{\text{SI}}(w_1, w_2) = \frac{2 \cdot \sigma_{\text{SI}}(w_1, w_2)}{\sigma_{\text{SI}}(w_1, w_1) + \sigma_{\text{SI}}(w_2, w_2)} \quad (7)$$

Note that T_{SI} is just the special case of Tversky's ratio model (5) in which $\alpha = 0.5$ and the similarity measure is symmetric.

We define three SI operations σ_{PROD}^2 , σ_{MIN} , and σ_{AVG} as follows:

SI	$\sigma_{\text{SI}}(w_1, w_2)$
PROD	$\sum_{f \in w_1 \cap w_2} w_1[f] * w_2[f]$
AVG	$\sum_{f \in w_1 \cap w_2} \frac{w_1[f] + w_2[f]}{2}$
MIN	$\sum_{f \in w_1 \cap w_2} \text{MIN}(w_1[f], w_2[f])$

¹Paralleling (7) is Jaccard-family normalization:

$$\sigma_{\text{JACC}}(w_1, w_2) = \frac{\sigma(w_1, w_2)}{\sigma(w_1, w_1) + \sigma(w_2, w_2) - \sigma(w_1, w_2)}$$

It is easy to generalize the result from van Rijsbergen (1979) for the original set-specific versions of Dice and Jaccard, and show that all of the Tversky family functions discussed above are monotonic in Jaccard.

² σ_{PROD} , of course, is dot product.

This yields the three similarity functions cited above:

$$\begin{aligned} \text{DICE PROD}(w_1, w_2) &= T_{\text{PROD}}(w_1, w_2) & (8) \\ \text{DICE}^\dagger(w_1, w_2) &= T_{\text{MIN}}(w_1, w_2) \\ \text{LIN}(w_1, w_2) &= T_{\text{AVG}}(w_1, w_2) \end{aligned}$$

Thus, all three of these functions are special cases of symmetric ratio models. Below, we investigate asymmetric versions of all three, which we write as $T_{\alpha, \text{SI}}(w_1, w_2)$, defined as:

$$\frac{\sigma_{\text{SI}}(w_1, w_2)}{\alpha \cdot \sigma_{\text{SI}}(w_1, w_1) + (1 - \alpha) \cdot \sigma_{\text{SI}}(w_2, w_2)} \quad (9)$$

Following Lee (1997), who investigates a different family of asymmetric similarity functions, we will refer to these as α -skewed measures.

We also will look at a **rank-biased** family of measures:

$$\begin{aligned} R_{\alpha, \text{SI}}(w_1, w_2) &= T_{\alpha, \text{SI}}(w_h, w_l) \\ \text{where } w_l &= \arg \min_{w \in \{w_1, w_2\}} \text{Rank}(w) \\ w_h &= \arg \max_{w \in \{w_1, w_2\}} \text{Rank}(w) \end{aligned} \quad (10)$$

Here, $T_{\alpha, \text{SI}}(w_h, w_l)$ is as defined in (9), and the α -weighted word is always the less frequent word. For example, consider comparing the 100-feature vector for *dinghy* to the 1000 feature vector for *boat*: if α is high, we give more weight to the proportion of *dinghy*'s features that are shared than we give to the proportion of *boat*'s features that are shared.

In the following sections we present data showing that the performance of a dependency-based similarity system in capturing human similarity judgments can be greatly improved with rank-bias and α -skewing. We will investigate the three asymmetric functions defined above.³ We argue that the advantages of rank bias are tied to improved similarity estimation when comparing vectors of very different dimensionality. We then turn to the problem of finding a word's nearest semantic neighbors. The nearest neighbor problem is a rather a natural ground in which to try out ideas on asymmetry, since the nearest neighbor relation is itself not symmetrical. We show that α -skewing can be used to improve the quality of nearest neighbors found for both high- and mid-frequency words.

³Interestingly, Equation (9) does not yield an asymmetric version of cosine. Plugging unit vectors into the α -skewed version of DICE PROD still leaves us with a symmetric function (COS), whatever the value of α .

2 Systems

1. We parsed the BNC with the Malt Dependency parser (Nivre, 2003) and the Stanford parser (Klein and Manning, 2003), creating two dependency DBs, using basically the design in Lin (1998), with features weighted by PMI (Church and Hanks, 1990).
2. For each of the 3 rank-biased similarity systems ($R_{\alpha, \text{SI}}$) and cosine, we computed correlations with human judgments for the pairs in 2 standard wordsets: the combined Miller-Charles/Rubenstein-Goodenough word sets (Miller and Charles, 1991; Rubenstein and Goodenough, 1965) and the Wordsim 353 word set (Finkelstein et al., 2002), as well as to a subset of the Wordsim set restricted to reflect semantic similarity judgments, which we will refer to as Wordsim 201.
3. For each of 3 α -skewed similarity systems ($T_{\alpha, \text{SI}}$) and cosine, we found the nearest neighbor from among BNC nouns (of any rank) for the 10,000 most frequent BNC nouns using the the dependency DB created in step 2.
4. To evaluate of the quality of the nearest neighbors pairs found in Step 4, we scored them using the Wordnet-based Personalized Pagerank system described in Agirre (2009) (UKB), a non distributional WordNet based measure, and the best system in Table 1.

3 Human correlations

Table 1 presents the Spearman's correlation with human judgments for Cosine, UKB, and our 3 α -skewed models using Malt-parser based vectors applied to the combined Miller-Charles/Rubenstein-Goodenough word sets, the Wordsim 353 word set, and the Wordsim 202 word set.

The first of each of the column pairs is a symmetric system, and the second a rank-biased variant, based on Equation (10). In all cases, the biased system improves on the performance of its symmetric counterpart; in the case of DICE[†] and DICE PROD, that improvement is enough for the biased system to outperform cosine, the best of the symmetric distributionally based systems. The value .97 was chosen for α because it produced the best α -system on the MC/RG corpus. That value

		MC/RG		Wdsm201		Wdsm353	
		$\alpha = .5$	$\alpha = .97$	$\alpha = .5$	$\alpha = .97$	$\alpha = .5$	$\alpha = .97$
Dice	DICE PROD	.59	.71	.50	.60	.35	.44
	LIN	.48	.62	.42	.54	.29	.39
	DICE [†]	.58	.67	.49	.58	.34	.43
Euc	Cosine	.65	NA	.56	NA	.41	NA
WN	UKB WN	.80	NA	.75	NA	.68	NA

Table 1: System/Human correlations. Above the line: MALT Parser-based systems

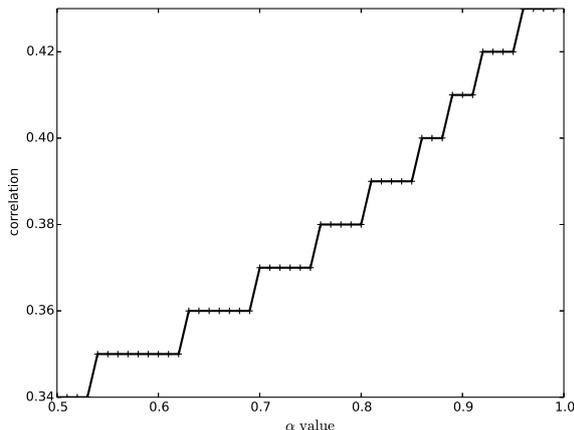


Figure 1: Scores monotonically increase with α

is probably probably an overtrained optimum. The point is that α -skewing always helps: For all three systems, the improvement shown in raising α from .5 to whatever the optimum is is monotonic. This is shown in Figure 1. Table 2 shows very similar results using the Stanford parser, demonstrating the pattern is not limited to a single parsing model.

In Table 3, we list the pairs whose reranking on the MC/RG dataset contributed most to the improvement of the $\alpha = .9$ system over the default $\alpha = .5$ system. In the last column an approximation of the amount of correlation improvement provided by that pair (δ):⁴ Note the 3 of the 5 items contributing the most improvement this system were pairs with a large difference in rank. Choosing $\alpha = .9$, weights recall toward the rarer word. We conjecture that the reason this helps is Tversky’s principle: It is natural to use the sparser

⁴The approximation is based on the formula for computing Spearman’s R with no ties. If n is the number of items, then the improvement on that item is:

$$\frac{6 * [(baseline - gold)^2 - (test - gold)^2]}{n * (n^2 - 1)}$$

Word 1	Rank	Word 2	Rank	δ
automobile	7411	car	100	0.030
asylum	3540	madhouse	14703	0.020
coast	708	hill	949	0.018
mound	3089	stove	2885	0.017
autograph	10136	signature	2743	0.009

Table 3: Pairs contributing the biggest improvement, MC/RG word set

representation as the focus in the comparison.

4 Nearest neighbors

Figure 2 gives the results of our nearest neighbor study on the BNC for the case of DICE PROD. The graphs for the other two α -skewed systems are nearly identical, and are not shown due to space limitations. The target word, the word whose nearest neighbor is being found, always receives the weight $1 - \alpha$. The x-axis shows target word rank; the y-axis shows the average UKB similarity scores assigned to nearest neighbors every 50 ranks. All the systems show degraded nearest neighbor quality as target words grow rare, but at lower ranks, the $\alpha = .04$ nearest neighbor system fares considerably better than the symmetric $\alpha = .50$ system; the line across the bottom tracks the score of a system with randomly generated nearest neighbors. The symmetric DICE PROD system is as an excellent nearest neighbor system at high ranks but drops below the $\alpha = .04$ system at around rank 3500. We see that the $\alpha = .8$ system is even better than the symmetric system at high ranks, but degrades much more quickly.

We explain these results on the basis of the principle developed for the human correlation data: To reflect natural judgments of similarity for comparisons of representations of differing sparseness, α should be tipped toward the sparser representation.

Thus, $\alpha = .80$ works best for high rank target words, because most nearest neighbor candi-

	MC/RG			Wdsm201			Wdsm353		
	$\alpha = .5$	opt	opt α	$\alpha = .5$	opt	opt α	$\alpha = .5$	opt	opt α
DICE PROD	.65	.70	.86	.42	.57	.99	.36	.44	.98
LIN	.58	.68	.90	.41	.56	.94	.30	.41	.99
DICE [†]	.60	.71	.91	.43	.53	.99	.32	.43	.99

Table 2: System/Human correlations for Stanford parser systems

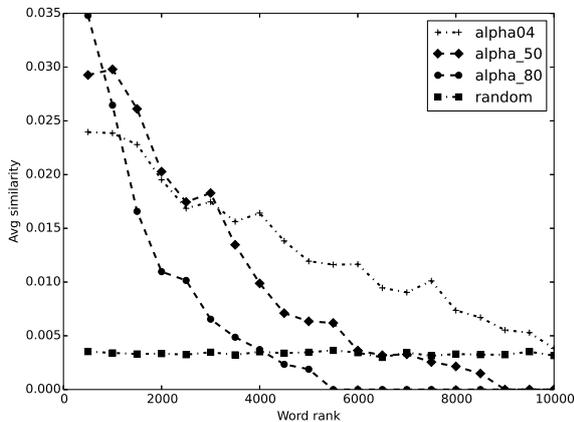


Figure 2: UKB evaluation scores for nearest neighbor pairs across word ranks, sampled every 50 ranks.

dates are less frequent, and $\alpha = .8$ tips the balance toward the nontarget words. On the other hand, when the target word is a low ranking word, a high α weight means it never receives the highest weight, and this is disastrous, since most good candidates are higher ranking. Conversely, $\alpha = .04$ works better.

5 Previous work

The debt owed to Tversky (1977) has been made clear in the introduction. Less clear is the debt owed to Jimenez et al. (2012), which also proposes an asymmetric similarity framework based on Tversky’s insights. Jimenez et al. showed the continued relevance of Tversky’s work.

Motivated by the problem of measuring how well the distribution of one word w_1 captures the distribution of another w_2 , Weeds and Weir (2005) also explore asymmetric models, expressing similarity calculations as weighted combinations of several variants of what they call precision and recall. Some of their models are also Tverskyan ratio models. To see this, we divide (9) everywhere by $\sigma(w_1, w_2)$:

$$T_{SI}(w_1, w_2) = \frac{1}{\frac{\alpha \cdot \sigma(w_1, w_1)}{\sigma(w_1, w_2)} + \frac{(1-\alpha) \cdot \sigma(w_2, w_2)}{\sigma(w_1, w_2)}}$$

If the SI is MIN, then the two terms in the denominator are the inverses of what W&W call difference-weighted precision and recall:

$$\begin{aligned} \text{PREC}(w_1, w_2) &= \frac{\sigma_{\text{MIN}}(w_1, w_2)}{\sigma_{\text{MIN}}(w_1, w_1)} \\ \text{REC}(w_1, w_2) &= \frac{\sigma_{\text{MIN}}(w_1, w_2)}{\sigma_{\text{MIN}}(w_2, w_2)}, \end{aligned}$$

So for T_{MIN} , (9) can be rewritten:

$$\frac{1}{\frac{\alpha}{\text{PREC}(w_1, w_2)} + \frac{1-\alpha}{\text{REC}(w_1, w_2)}}$$

That is, T_{MIN} is a weighted harmonic mean of precision and recall, the so-called weighted F-measure (Manning and Schütze, 1999). W&W’s additive precision/recall models appear not to be Tversky models, since they compute separate sums for precision and recall from the $f \in w_1 \cap w_2$, one using $w_1[f]$, and one using $w_2[f]$.

Long before Weeds and Weir, Lee (1999) proposed an asymmetric similarity measure as well. Like Weeds and Weir, her perspective was to calculate the effectiveness of using one distribution as a proxy for the other, a fundamentally asymmetric problem. For distributions q and r , Lee’s α -skew divergence takes the KL-divergence of a mixture of q and r from q , using the α parameter to define the proportions in the mixture.

6 Conclusion

We have shown that Tversky’s asymmetric ratio models can improve performance in capturing human judgments and produce better nearest neighbors. To validate these very preliminary results, we need to explore applications compatible with asymmetry, such as the TOEFL-like synonym discovery task in Freitag et al. (2005), and the PP-attachment task in Dagan et al. (1999).

Acknowledgments

This work reported here was supported by NSF CDI grant # 1028177.

References

- E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Pasca, and A. Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of NAACL-HLT 09*, Boulder, Co.
- K.W. Church and P. Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.
- J.R. Curran. 2004. *From Distributional to Semantic Similarity*. Ph.D. thesis, University of Edinburgh. College of Science and Engineering. School of Informatics.
- I. Dagan, L. Lee, and F.C.N. Pereira. 1999. Similarity-based models of word cooccurrence probabilities. *Machine Learning*, 34(1):43–69.
- L.R. Dice. 1945. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302.
- L. Finkelstein, E. Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Rupp. 2002. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.
- D. Freitag, M. Blume, J. Byrnes, E. Chow, S. Kapadia, R. Rohwer, and Z. Wang. 2005. New experiments in distributional representations of synonymy. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 25–32. Association for Computational Linguistics.
- R. L. Goldstone. in press. Similarity. In R.A. Wilson Wilson and F. C. Keil, editors, *MIT Encyclopedia of Cognitive Sciences*. MIT Press, Cambridge, MA.
- S. Jimenez, C. Becerra, and A. Gelbukh. 2012. Soft cardinality: A parameterized similarity function for text comparison. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 449–453. Association for Computational Linguistics.
- D. Klein and Christopher D. Manning. 2003. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, pages 3–10, Cambridge, MA. MIT Press.
- L. Lee. 1997. *Similarity-based approaches to natural language processing*. Ph.D. thesis, Harvard University.
- L. Lee. 1999. Measures of distributional similarity. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 25–32. Association for Computational Linguistics.
- D. Lin. 1998. Automatic retrieval and clustering of similar words. In *Annual Meeting-Association for Computational Linguistics*, volume 36, pages 768–774. Association for Computational Linguistics.
- C.D. Manning and H. Schütze. 1999. *Foundations of statistical natural language processing*. MIT Press, Cambridge.
- G.A. Miller and W.G. Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28.
- J. Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT 03)*, pages 149–160.
- E. Rosch and C. B. Mervis. 1975. Family resemblances: Studies in the internal structure of categories. *Cognitive psychology*, 7(4):573–605.
- H. Rubenstein and J.B. Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8:627–633.
- A. Tversky. 1977. Features of similarity. *Psychological Review*, 84:327–352.
- C. J. van Rijsbergen. 1979. *Information retrieval*. Butterworth-Heinemann, Oxford.
- J. Weeds and D. Weir. 2005. Co-occurrence retrieval: A flexible framework for lexical distributional similarity. *Computational Linguistics*, 31(4):439–475.

Dependency-Based Word Embeddings

Omer Levy* and Yoav Goldberg

Computer Science Department

Bar-Ilan University

Ramat-Gan, Israel

{omerlevy, yoav.goldberg}@gmail.com

Abstract

While continuous word embeddings are gaining popularity, current models are based solely on linear contexts. In this work, we generalize the skip-gram model with negative sampling introduced by Mikolov et al. to include arbitrary contexts. In particular, we perform experiments with dependency-based contexts, and show that they produce markedly different embeddings. The dependency-based embeddings are less topical and exhibit more functional similarity than the original skip-gram embeddings.

1 Introduction

Word representation is central to natural language processing. The default approach of representing words as discrete and distinct symbols is insufficient for many tasks, and suffers from poor generalization. For example, the symbolic representation of the words “pizza” and “hamburger” are completely unrelated: even if we know that the word “pizza” is a good argument for the verb “eat”, we cannot infer that “hamburger” is also a good argument. We thus seek a representation that captures semantic and syntactic similarities between words. A very common paradigm for acquiring such representations is based on the distributional hypothesis of Harris (1954), stating that words in similar contexts have similar meanings.

Based on the distributional hypothesis, many methods of deriving word representations were explored in the NLP community. On one end of the spectrum, words are grouped into clusters based on their contexts (Brown et al., 1992; Uszkor-eit and Brants, 2008). On the other end, words

are represented as a very high dimensional but sparse vectors in which each entry is a measure of the association between the word and a particular context (see (Turney and Pantel, 2010; Baroni and Lenci, 2010) for a comprehensive survey). In some works, the dimensionality of the sparse word-context vectors is reduced, using techniques such as SVD (Bullinaria and Levy, 2007) or LDA (Ritter et al., 2010; Séaghdha, 2010; Cohen et al., 2012). Most recently, it has been proposed to represent words as dense vectors that are derived by various training methods inspired from neural-network language modeling (Bengio et al., 2003; Collobert and Weston, 2008; Mnih and Hinton, 2008; Mikolov et al., 2011; Mikolov et al., 2013b). These representations, referred to as “neural embeddings” or “word embeddings”, have been shown to perform well across a variety of tasks (Turian et al., 2010; Collobert et al., 2011; Socher et al., 2011; Al-Rfou et al., 2013).

Word embeddings are easy to work with because they enable efficient computation of word similarities through low-dimensional matrix operations. Among the state-of-the-art word-embedding methods is the *skip-gram with negative sampling* model (SKIPGRAM), introduced by Mikolov et al. (2013b) and implemented in the `word2vec` software.¹ Not only does it produce useful word representations, but it is also very efficient to train, works in an online fashion, and scales well to huge corpora (billions of words) as well as very large word and context vocabularies.

Previous work on neural word embeddings take the contexts of a word to be its *linear context* – words that precede and follow the target word, typically in a window of k tokens to each side. However, other types of contexts can be explored too.

In this work, we generalize the SKIPGRAM model, and move from linear bag-of-words contexts to arbitrary word contexts. Specifically,

*Supported by the European Community’s Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 287923 (EXCITEMENT).

¹code.google.com/p/word2vec/

following work in sparse vector-space models (Lin, 1998; Padó and Lapata, 2007; Baroni and Lenci, 2010), we experiment with *syntactic contexts* that are derived from automatically produced dependency parse-trees.

The different kinds of contexts produce noticeably different embeddings, and induce different word similarities. In particular, the bag-of-words nature of the contexts in the “original” SKIPGRAM model yield *broad topical similarities*, while the dependency-based contexts yield more *functional* similarities of a *cohyponym* nature. This effect is demonstrated using both qualitative and quantitative analysis (Section 4).

The neural word-embeddings are considered opaque, in the sense that it is hard to assign meanings to the dimensions of the induced representation. In Section 5 we show that the SKIPGRAM model does allow for some introspection by querying it for contexts that are “activated by” a target word. This allows us to peek into the learned representation and explore the contexts that are found by the learning process to be most discriminative of particular words (or groups of words). To the best of our knowledge, this is the first work to suggest such an analysis of discriminatively-trained word-embedding models.

2 The Skip-Gram Model

Our departure point is the skip-gram neural embedding model introduced in (Mikolov et al., 2013a) trained using the negative-sampling procedure presented in (Mikolov et al., 2013b). In this section we summarize the model and training objective following the derivation presented by Goldberg and Levy (2014), and highlight the ease of incorporating arbitrary contexts in the model.

In the skip-gram model, each word $w \in W$ is associated with a vector $v_w \in R^d$ and similarly each context $c \in C$ is represented as a vector $v_c \in R^d$, where W is the words vocabulary, C is the contexts vocabulary, and d is the embedding dimensionality. The entries in the vectors are latent, and treated as parameters to be learned. Loosely speaking, we seek parameter values (that is, vector representations for both words and contexts) such that the dot product $v_w \cdot v_c$ associated with “good” word-context pairs is maximized.

More specifically, the negative-sampling objective assumes a dataset D of observed (w, c) pairs of words w and the contexts c , which appeared in

a large body of text. Consider a word-context pair (w, c) . Did this pair come from the data? We denote by $p(D = 1|w, c)$ the probability that (w, c) came from the data, and by $p(D = 0|w, c) = 1 - p(D = 1|w, c)$ the probability that (w, c) did not. The distribution is modeled as:

$$p(D = 1|w, c) = \frac{1}{1 + e^{-v_w \cdot v_c}}$$

where v_w and v_c (each a d -dimensional vector) are the model parameters to be learned. We seek to maximize the log-probability of the observed pairs belonging to the data, leading to the objective:

$$\arg \max_{v_w, v_c} \sum_{(w, c) \in D} \log \frac{1}{1 + e^{-v_w \cdot v_c}}$$

This objective admits a trivial solution in which $p(D = 1|w, c) = 1$ for every pair (w, c) . This can be easily achieved by setting $v_c = v_w$ and $v_c \cdot v_w = K$ for all c, w , where K is large enough number.

In order to prevent the trivial solution, the objective is extended with (w, c) pairs for which $p(D = 1|w, c)$ must be low, i.e. pairs which are not in the data, by generating the set D' of random (w, c) pairs (assuming they are all incorrect), yielding the negative-sampling training objective:

$$\arg \max_{v_w, v_c} \left(\prod_{(w, c) \in D} p(D = 1|c, w) \prod_{(w, c) \in D'} p(D = 0|c, w) \right)$$

which can be rewritten as:

$$\arg \max_{v_w, v_c} \left(\sum_{(w, c) \in D} \log \sigma(v_c \cdot v_w) + \sum_{(w, c) \in D'} \log \sigma(-v_c \cdot v_w) \right)$$

where $\sigma(x) = 1/(1 + e^x)$. The objective is trained in an online fashion using stochastic-gradient updates over the corpus $D \cup D'$.

The negative samples D' can be constructed in various ways. We follow the method proposed by Mikolov et al.: for each $(w, c) \in D$ we construct n samples $(w, c_1), \dots, (w, c_n)$, where n is a hyperparameter and each c_j is drawn according to its unigram distribution raised to the $3/4$ power.

Optimizing this objective makes observed word-context pairs have similar embeddings, while scattering unobserved pairs. Intuitively, words that appear in similar contexts should have similar embeddings, though we have not yet found a formal proof that SKIPGRAM does indeed maximize the dot product of similar words.

3 Embedding with Arbitrary Contexts

In the SKIPGRAM embedding algorithm, the contexts of a word w are the words surrounding it

in the text. The context vocabulary C is thus identical to the word vocabulary W . However, this restriction is not required by the model; contexts need not correspond to words, and the number of context-types can be substantially larger than the number of word-types. We generalize SKIPGRAM by replacing the bag-of-words contexts with arbitrary contexts.

In this paper we experiment with dependency-based *syntactic contexts*. Syntactic contexts capture different information than bag-of-word contexts, as we demonstrate using the sentence “*Australian scientist discovers star with telescope*”.

Linear Bag-of-Words Contexts This is the context used by `word2vec` and many other neural embeddings. Using a window of size k around the target word w , $2k$ contexts are produced: the k words before and the k words after w . For $k = 2$, the contexts of the target word w are $w_{-2}, w_{-1}, w_{+1}, w_{+2}$. In our example, the contexts of *discovers* are *Australian, scientist, star, with*.²

Note that a context window of size 2 may miss some important contexts (*telescope* is not a context of *discovers*), while including some accidental ones (*Australian* is a context *discovers*). Moreover, the contexts are unmarked, resulting in *discovers* being a context of both *stars* and *scientists*, which may result in *stars* and *scientists* ending up as neighbours in the embedded space. A window size of 5 is commonly used to capture broad topical content, whereas smaller windows contain more focused information about the target word.

Dependency-Based Contexts An alternative to the bag-of-words approach is to derive contexts based on the syntactic relations the word participates in. This is facilitated by recent advances in parsing technology (Goldberg and Nivre, 2012; Goldberg and Nivre, 2013) that allow parsing to syntactic dependencies with very high speed and near state-of-the-art accuracy.

After parsing each sentence, we derive word contexts as follows: for a target word w with modifiers m_1, \dots, m_k and a head h , we consider the contexts $(m_1, lbl_1), \dots, (m_k, lbl_k), (h, lbl_h^{-1})$,

²`word2vec`’s implementation is slightly more complicated. The software defaults to prune rare words based on their frequency, and has an option for sub-sampling the frequent words. These pruning and sub-sampling happen *before* the context extraction, leading to a dynamic window size. In addition, the window size is not fixed to k but is sampled uniformly in the range $[1, k]$ for each word.

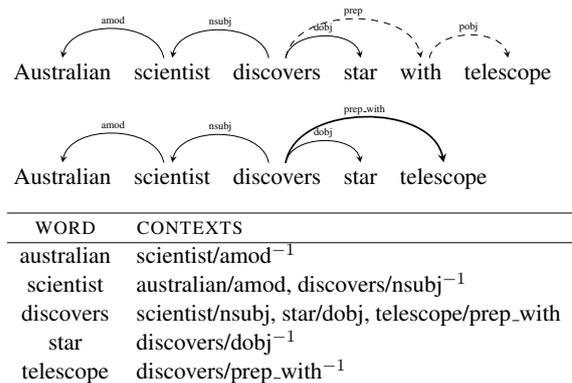


Figure 1: Dependency-based context extraction example. **Top:** preposition relations are collapsed into single arcs, making *telescope* a direct modifier of *discovers*. **Bottom:** the contexts extracted for each word in the sentence.

where lbl is the type of the dependency relation between the head and the modifier (e.g. *nsubj*, *dobj*, *prep_with*, *amod*) and lbl^{-1} is used to mark the inverse-relation. Relations that include a preposition are “collapsed” prior to context extraction, by directly connecting the head and the object of the preposition, and subsuming the preposition itself into the dependency label. An example of the dependency context extraction is given in Figure 1.

Notice that syntactic dependencies are both more inclusive and more focused than bag-of-words. They capture relations to words that are far apart and thus “out-of-reach” with small window bag-of-words (e.g. the instrument of *discover* is *telescope/prep_with*), and also filter out “coincidental” contexts which are within the window but not directly related to the target word (e.g. *Australian* is not used as the context for *discovers*). In addition, the contexts are typed, indicating, for example, that *stars* are objects of discovery and *scientists* are subjects. We thus expect the syntactic contexts to yield more focused embeddings, capturing more functional and less topical similarity.

4 Experiments and Evaluation

We experiment with 3 training conditions: BOW5 (bag-of-words contexts with $k = 5$), BOW2 (same, with $k = 2$) and DEPS (dependency-based syntactic contexts). We modified `word2vec` to support arbitrary contexts, and to output the context embeddings in addition to the word embeddings. For bag-of-words contexts we used the original `word2vec` implementation, and for syntactic contexts, we used our modified version. The negative-sampling parameter (how many negative contexts to sample for every correct one) was 15.

All embeddings were trained on English Wikipedia. For DEPS, the corpus was tagged with parts-of-speech using the Stanford tagger (Toutanova et al., 2003) and parsed into labeled Stanford dependencies (de Marneffe and Manning, 2008) using an implementation of the parser described in (Goldberg and Nivre, 2012). All tokens were converted to lowercase, and words and contexts that appeared less than 100 times were filtered. This resulted in a vocabulary of about 175,000 words, with over 900,000 distinct syntactic contexts. We report results for 300 dimension embeddings, though similar trends were also observed with 600 dimensions.

4.1 Qualitative Evaluation

Our first evaluation is qualitative: we manually inspect the 5 most similar words (by cosine similarity) to a given set of target words (Table 1).

The first target word, *Batman*, results in similar sets across the different setups. This is the case for many target words. However, other target words show clear differences between embeddings.

In *Hogwarts* - the school of magic from the fictional Harry Potter series - it is evident that BOW contexts reflect the *domain* aspect, whereas DEPS yield a list of famous schools, capturing the *semantic type* of the target word. This observation holds for *Turing*³ and many other nouns as well; BOW find words that *associate* with *w*, while DEPS find words that *behave* like *w*. Turney (2012) described this distinction as *domain similarity* versus *functional similarity*.

The *Florida* example presents an ontological difference; bag-of-words contexts generate meronyms (counties or cities within Florida), while dependency-based contexts provide cohyponyms (other US states). We observed the same behavior with other geographical locations, particularly with countries (though not all of them).

The next two examples demonstrate that similarities induced from DEPS share a syntactic function (adjectives and gerunds), while similarities based on BOW are more diverse. Finally, we observe that while both BOW5 and BOW2 yield topical similarities, the larger window size result in more topicality, as expected.

³DEPS generated a list of scientists whose name ends with “ing”. This is may be a result of occasional POS-tagging errors. Still, the embedding does a remarkable job and retrieves scientists, despite the noisy POS. The list contains more mathematicians without “ing” further down.

Target Word	BOW5	BOW2	DEPS
batman	nightwing aquaman catwoman superman manhunter	superman superboy aquaman catwoman batgirl	superman superboy supergirl catwoman aquaman
hogwarts	dumbledore hallows half-blood malfoy snape	evernight sunnydale garderobe blandings collinwood	sunnydale collinwood calarts greendale millfield
turing	nondeterministic non-deterministic computability deterministic finite-state	non-deterministic finite-state nondeterministic buchi primality	pauling hottelling heting lessing hamming
florida	gainesville fla jacksonville tampa lauderdale	fla alabama gainesville tallahassee texas	texas louisiana georgia california carolina
object-oriented	aspect-oriented smalltalk event-driven prolog domain-specific	aspect-oriented event-driven objective-c dataflow 4gl	event-driven domain-specific rule-based data-driven human-centered
dancing	singing dance dances dancers tap-dancing	singing dance dances breakdancing clowning	singing rapping breakdancing miming busking

Table 1: Target words and their 5 most similar words, as induced by different embeddings.

We also tried using the subsampling option (Mikolov et al., 2013b) with BOW contexts (not shown). Since `word2vec` removes the subsampled words from the corpus *before* creating the window contexts, this option effectively increases the window size, resulting in greater topicality.

4.2 Quantitative Evaluation

We supplement the examples in Table 1 with quantitative evaluation to show that the qualitative differences pointed out in the previous section are indeed widespread. To that end, we use the WordSim353 dataset (Finkelstein et al., 2002; Agirre et al., 2009). This dataset contains pairs of similar words that reflect either *relatedness* (topical similarity) or *similarity* (functional similarity) relations.⁴ We use the embeddings in a retrieval/ranking setup, where the task is to rank the *similar* pairs in the dataset above the *related* ones.

The pairs are ranked according to cosine similarities between the embedded words. We then draw a recall-precision curve that describes the embedding’s affinity towards one subset (“similarity”) over another (“relatedness”). We expect DEPS’s curve to be higher than BOW2’s curve, which in turn is expected to be higher than

⁴Some word pairs are judged to exhibit both types of similarity, and were ignored in this experiment.

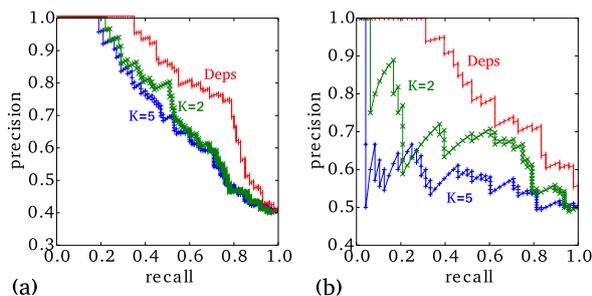


Figure 2: Recall-precision curve when attempting to rank the *similar* words above the *related* ones. (a) is based on the WordSim353 dataset, and (b) on the Chiarello et al. dataset.

BOW’s. The graph in Figure 2a shows this is indeed the case. We repeated the experiment with a different dataset (Chiarello et al., 1990) that was used by Turney (2012) to distinguish between domain and functional similarities. The results show a similar trend (Figure 2b). When reversing the task such that the goal is to rank the *related* terms above the *similar* ones, the results are reversed, as expected (not shown).⁵

5 Model Introspection

Neural word embeddings are often considered opaque and uninterpretable, unlike sparse vector space representations in which each dimension corresponds to a particular known context, or LDA models where dimensions correspond to latent topics. While this is true to a large extent, we observe that SKIPGRAM does allow a non-trivial amount of introspection. Although we cannot assign a meaning to any particular dimension, we can indeed get a glimpse at the kind of information being captured by the model, by examining which contexts are “activated” by a target word.

Recall that the learning procedure is attempting to maximize the dot product $v_c \cdot v_w$ for good (w, c) pairs and minimize it for bad ones. If we keep the context embeddings, we can query the model for the contexts that are most activated by (have the highest dot product with) a given target word. By doing so, we can see what the model learned to be a good discriminative context for the word.

To demonstrate, we list the 5 most activated contexts for our example words with DEPS embeddings in Table 2. Interestingly, the most discriminative syntactic contexts in these cases are

⁵Additional experiments (not presented in this paper) reinforce our conclusion. In particular, we found that DEPS perform dramatically worse than BOW contexts on analogy tasks as in (Mikolov et al., 2013c; Levy and Goldberg, 2014).

batman	hogwarts	turing
superman/conj ⁻¹	students/prep_at ⁻¹	machine/nn ⁻¹
spider-man/conj ⁻¹	educated/prep_at ⁻¹	test/nn ⁻¹
superman/conj	student/prep_at ⁻¹	theorem/poss ⁻¹
spider-man/conj	stay/prep_at ⁻¹	machines/nn ⁻¹
robin/conj	learned/prep_at ⁻¹	tests/nn ⁻¹
florida	object-oriented	dancing
marlins/nn ⁻¹	programming/amod ⁻¹	dancing/conj
beach/appos ⁻¹	language/amod ⁻¹	dancing/conj ⁻¹
jacksonville/appos ⁻¹	framework/amod ⁻¹	singing/conj ⁻¹
tampa/appos ⁻¹	interface/amod ⁻¹	singing/conj
florida/conj ⁻¹	software/amod ⁻¹	ballroom/nn

Table 2: Words and their top syntactic contexts.

not associated with subjects or objects of verbs (or their inverse), but rather with conjunctions, appositions, noun-compounds and adjectival modifiers. Additionally, the collapsed preposition relation is very useful (e.g. for capturing the *school* aspect of *hogwarts*). The presence of many conjunction contexts, such as *superman/conj* for *batman* and *singing/conj* for *dancing*, may explain the functional similarity observed in Section 4; conjunctions in natural language tend to enforce their conjuncts to share the same semantic types and inflections.

In the future, we hope that insights from such model introspection will allow us to develop better contexts, by focusing on conjunctions and prepositions for example, or by trying to figure out why the subject and object relations are absent and finding ways of increasing their contributions.

6 Conclusions

We presented a generalization of the SKIPGRAM embedding model in which the linear bag-of-words contexts are replaced with arbitrary ones, and experimented with dependency-based contexts, showing that they produce markedly different kinds of similarities. These results are expected, and follow similar findings in the distributional semantics literature. We also demonstrated how the resulting embedding model can be queried for the discriminative contexts for a given word, and observed that the learning procedure seems to favor relatively local syntactic contexts, as well as conjunctions and objects of preposition. We hope these insights will facilitate further research into improved context modeling and better, possibly task-specific, embedded representations. Our software, allowing for experimentation with arbitrary contexts, together with the embeddings described in this paper, are available for download at the authors’ websites.

References

- Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27, Boulder, Colorado, June. Association for Computational Linguistics.
- Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual nlp. In *Proc. of CoNLL 2013*.
- Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.
- Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.
- Peter F Brown, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural. *Computational Linguistics*, 18(4).
- John A Bullinaria and Joseph P Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39(3):510–526.
- Christine Chiarello, Curt Burgess, Lorie Richards, and Alma Pollock. 1990. Semantic and associative priming in the cerebral hemispheres: Some words do, some words don't... sometimes, some places. *Brain and Language*, 38(1):75–104.
- Raphael Cohen, Yoav Goldberg, and Michael Elhadad. 2012. Domain adaptation of a dependency parser with a class-class selectional preference model. In *Proceedings of ACL 2012 Student Research Workshop*, pages 43–48, Jeju Island, Korea, July. Association for Computational Linguistics.
- Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.
- Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, Manchester, UK, August. Coling 2008 Organizing Committee.
- Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppín. 2002. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.
- Yoav Goldberg and Omer Levy. 2014. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.
- Yoav Goldberg and Joakim Nivre. 2012. A dynamic oracle for the arc-eager system. In *Proc. of COLING 2012*.
- Yoav Goldberg and Joakim Nivre. 2013. Training deterministic parsers with non-deterministic oracles. *Transactions of the association for Computational Linguistics*, 1.
- Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.
- 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*, Baltimore, Maryland, USA, June. Association for Computational Linguistics.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 2, ACL '98*, pages 768–774, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Tomas Mikolov, Stefan Kombrink, Lukas Burget, JH Cernocky, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5528–5531. IEEE.
- 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, Gregory S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119.
- 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, Atlanta, Georgia, June. Association for Computational Linguistics.

- Andriy Mnih and Geoffrey E Hinton. 2008. A scalable hierarchical distributed language model. In *Advances in Neural Information Processing Systems*, pages 1081–1088.
- Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.
- Alan Ritter, Mausam, and Oren Etzioni. 2010. A latent dirichlet allocation method for selectional preferences. In *ACL*, pages 424–434.
- Diarmuid Ó Séaghdha. 2010. Latent variable models of selectional preference. In *ACL*, pages 435–444.
- Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics.
- Kristina Toutanova, Dan Klein, Chris Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of NAACL*.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.
- P.D. Turney and P. Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.
- Peter D. Turney. 2012. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research*, 44:533–585.
- Jakob Uszkoreit and Thorsten Brants. 2008. Distributed word clustering for large scale class-based language modeling in machine translation. In *Proc. of ACL*, pages 755–762.

Vector spaces for historical linguistics: Using distributional semantics to study syntactic productivity in diachrony

Florent Perek

Princeton University

Princeton, NJ, USA

fperek@princeton.edu

Abstract

This paper describes an application of distributional semantics to the study of syntactic productivity in diachrony, i.e., the property of grammatical constructions to attract new lexical items over time. By providing an empirical measure of semantic similarity between words derived from lexical co-occurrences, distributional semantics not only reliably captures how the verbs in the distribution of a construction are related, but also enables the use of visualization techniques and statistical modeling to analyze the semantic development of a construction over time and identify the semantic determinants of syntactic productivity in naturally occurring data.

1 Introduction

Language change does not exclusively consist of drastic shifts in ‘core’ aspects of grammar, such as changes in word order. Variation in usage, which can occur in no more than a few decades, is much more common, and to many linguists constitutes linguistic change in the making. Among these aspects of language use that are subject to diachronic change, this paper is concerned with the productivity of syntactic constructions, i.e., the range of lexical items with which a construction can be used. A given construction might occur with very different distributions at different points in time, even when the function it conveys remains the same. This is what Israel (1996) finds for the pattern “Verb *one’s way* Path”, commonly called the *way*-construction (Goldberg, 1995), exemplified by (1) and (2) below.

- (1) They hacked their way through the jungle.
- (2) She typed her way to a promotion.

As reported by Israel, examples like (1), in which the main verb describes the physical means

whereby motion towards a goal is enabled, are attested as early as the 16th century, but it was not until the 19th century that examples like (2) started to appear, in which the action depicted by the verb provides a more indirect (and abstract) way of attaining the agent’s goal.

The productivity of a construction may appear partly arbitrary, but a growing body of evidence suggests that it is tied to the previous experience of speakers with that construction (Barðdal, 2008; Bybee and Eddington, 2006; Suttle and Goldberg, 2011). More specifically, previous research points to a strong semantic component, in that the possibility of a novel use depends on how it semantically relates to prior usage. Along these lines, Suttle and Goldberg (2011, 1254) posit a criterion of coverage, defined as “the degree to which attested instances ‘cover’ the category determined jointly by attested instances together with the target coinage”. Coverage relates to how the semantic domain of a construction is populated in the vicinity of a given target coinage, and in particular to the density of the semantic space.

The importance of semantics for syntactic productivity implies that the meaning of lexical items must be appropriately taken into account when studying the distribution of constructions, which calls for an empirical operationalization of semantics. Most existing studies rely either on the semantic intuitions of the analyst, or on semantic norming studies (Bybee and Eddington, 2006). In this paper, I present a third alternative that takes advantage of advances in computational linguistics and draws on a distributionally-based measure of semantic similarity. On the basis of a case study of the construction “V *the hell out of* NP”, I show how distributional semantics can profitably be applied to the study of syntactic productivity.

2 The *hell*-construction

The case study presented in this paper considers the syntactic pattern “V *the hell out of* NP”, as exemplified by the following sentences from the Corpus of Contemporary American English (COCA; Davies, 2008):

- (3) Snakes just scare the hell out of me.
- (4) It surprised the hell out of me when I heard what he’s been accused of.
- (5) You might kick the hell out of me like you did that doctor.

The construction generally conveys an intensifying function (very broadly defined). Thus, *scare/surprise the hell out of* means “scare/surprise very much”, and *kick the hell out of* means “kick very hard”. The particular aspect that is intensified may be highly specific to the verb and depend to some extent on the context. *Scare* and *beat* are the most typical verbs in that construction (and arguably the two that first come to mind), but a wide and diverse range of other verbs can also be found, such that *avoid* in (6), *drive* (a car) in (7) and even an intransitive verb (*listen*) in (8):

- (6) I [...] avoided the hell out of his presence.
- (7) But you drove the hell out of it!
- (8) I’ve been listening the hell out of your tape.

To examine how the construction evolved over time, I used diachronic data from the Corpus of Historical American English (COHA; Davies 2010), which contains about 20 million words of written American English for each decade between 1810 and 2009 roughly balanced for genre (fiction, magazines, newspapers, non-fiction). Instances of the *hell*-construction were filtered out manually from the results of the query “[v*] the hell out of”, mostly ruling out locative constructions like *get the hell out of here*. The diachronic evolution of the verb slot in terms of token and type frequency is plotted in Figure 1. Since the corpus size varies slightly in each decade, the token frequencies are normalized per million words.

The construction is first attested in the corpus in the 1930s. Since then, it has been steadily increasing in token frequency (to the exception of a sudden decrease in the 1990s). Also, more and more different verbs are attested in the construction, as shown by the increase in type frequency.

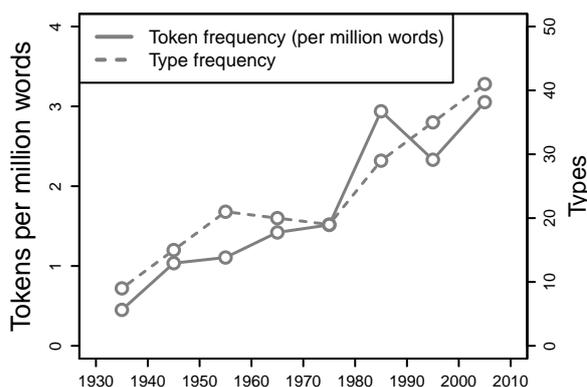


Figure 1: Diachronic development of the *hell*-construction in terms of normalized token frequency and type frequency

This reflects a general expansion of the productivity of the construction, but it does not show what this productivity consists of. For instance, it does not say what kinds of verbs joined the distribution and to what extent the distribution becomes semantically more diverse over time. To answer these questions, I will analyze the distribution of the construction from a semantic point of view by using a measure of semantic similarity derived from distributional information.

3 Distributional measure of semantic similarity

Drawing on the observation that words occurring in similar contexts tend to have related meanings (Miller and Charles, 1991), distributional approaches to semantics seek to capture the meaning of words through their distribution in large text corpora (Lenci, 2008; Turney and Pantel, 2010; Erk, 2012). One benefit of the distributional semantics approach is that it allows semantic similarity between words to be quantified by measuring the similarity in their distribution. This is achieved by means of a vector-space model that assigns an array of numerical values (i.e., a vector) derived from distributional information to each word. A wide range of distributional information can be employed in vector-based models; the present study uses the ‘bag of words’ approach, which is based on the frequency of co-occurrence of words within a given context window. According to Sahlgren (2008), this kind of model captures to what extent words can be substituted for each other, which is a good measure of semantic similarity between verbs. As it turns out, even this

relatively coarse model captures semantic distinctions in the distribution of the *hell*-construction that make intuitive sense.

All instances of the relevant verbs were extracted from the COCA¹ with their context of occurrence. In order to make sure that enough distributional information is available to reliably assess semantic similarity, verbs with less than 2,000 occurrences were excluded, which left 92 usable items (out of 105). The words in the sentence contexts extracted from the COCA were lemmatized and annotated for part-of-speech using TreeTagger (Schmid, 1994). The part-of-speech annotated lemma of each collocate within a 5-word window was extracted from the COCA data to build the co-occurrence matrix recording the frequency of co-occurrence of each verb with its collocates. Only the nouns, verbs, adjectives, and adverbs listed among the 5,000 most frequent words in the corpus were considered (to the exclusion of *be*, *have*, and *do*), thus ignoring function words (articles, prepositions, conjunctions, etc.) and all words that did not make the top 5,000.

The co-occurrence matrix was transformed by applying a Point-wise Mutual Information weighting scheme, using the DISSECT toolkit (Dinu et al., 2013), to turn the raw frequencies into weights that reflect how distinctive a collocate is for a given target word with respect to the other target words under consideration. The resulting matrix, which contains the distributional information (in 4,683 columns) for 92 verbs occurring in the *hell*-construction, constitutes the semantic space under consideration in this case study. Pairwise distances between the target verbs were calculated using the cosine distance. The rest of the analysis was conducted on the basis of this distance matrix in the R environment (R Development Core Team, 2013).

¹The COCA contains 464 million words of American English consisting of the same amount of spoken, fiction, magazine, newspaper, and academic prose data for each year between 1990 and 2012. Admittedly, a more ecologically valid choice would have been to use data from a particular time frame to build a vector-space model for the same time frame, but even the twenty-odd million words per decade of the COHA did not prove sufficient to achieve that purpose. This is, however, not as problematic as it might sound, since the meaning of the verbs under consideration are not likely to have changed considerably within the time frame of this study. Besides, using the same data presents the advantage that the distribution is modeled with the same semantic space in all time periods, which makes it easier to visualize changes.

4 Application of the vector-space model

4.1 Semantic plots

One of the advantages conferred by the quantification of semantic similarity is that lexical items can be precisely considered in relation to each other, and by aggregating the similarity information for all items in the distribution, we can produce a visual representation of the structure of the semantic domain of the construction in order to observe how verbs in that domain are related to each other, and to immediately identify the regions of the semantic space that are densely populated (with tight clusters of verbs), and those that are more sparsely populated (fewer and/or more scattered verbs). Multidimensional scaling (MDS) provides a way both to aggregate similarity information and to represent it visually. This technique aims to place objects in a space with two (or more) dimensions such that the between-object distances are preserved as much as possible.

The pairwise distances between verbs were submitted to multidimensional scaling into two dimensions.² To visualize the semantic development of the *hell*-construction over time, the diachronic data was divided into four successive twenty-year periods: 1930-1949, 1950-1969, 1970-1989, and 1990-2009. The semantic plots corresponding to the distribution of the construction in each period are presented in Figure 2. For convenience and ease of visualization, the verbs are color-coded according to four broad semantic groupings that were identified inductively by means of hierarchical clustering (using Ward's criterion).³

By comparing the plots in Figure 2, we can follow the semantic development of the *hell*-construction. The construction is strikingly centered around two kinds of verbs: mental verbs (in red: *surprise*, *please*, *scare*, etc.) and verbs of hitting (most verbs in green: *smash*, *kick*, *whack*, etc.), a group that is orbited by other kinds of forceful actions (such as *pinch*, *push*, and *tear*). These two types of verbs account for most of the distribution at the onset, and they continue to

²Non-metric MDS was employed (Kruskal, 1964), using the function `isoMDS` from the R package MASS.

³Another benefit of combining clustering and MDS stems from the fact that the latter often distorts the data when fitting the objects into two dimensions, in that some objects might have to be slightly misplaced if not all distance relations can be simultaneously complied with. Since cluster analysis operates with all 4,683 dimensions of the distributional space, it is more reliable than MDS, although it lacks the visual appeal of the latter.

and its n^{th} nearest neighbor. In plain language, density equals one minus the mean distance to the N nearest neighbors. The latter value decreases with space density (i.e., if there are many close neighbors), and is therefore technically a measure of sparsity; since cosine distances are between 0 and 1, subtracting the mean distance from one returns a measure of density within the same boundaries.

This measure of density was used as a factor in logistic regression to predict the first occurrence of a verb in the construction, coded as the binary variable OCCURRENCE, set to 1 for the first period in which the verb is attested in the construction, and to 0 for all preceding periods (later periods were discarded). For each VERB-PERIOD-OCCURRENCE triplet, the density of the semantic space around the verb in the immediately preceding period was calculated. Six different versions of the density measure, with the number of neighbors under consideration (N) varying between 3 and 8, were used to fit six mixed effects regression models with OCCURRENCE as the dependent variable, DENSITY as a fixed effect, and random by-verb intercepts and slopes (Bates et al., 2011). The results of these models are summarized in Table 1.

N	Effect of DENSITY	p -value
3	0.7211	0.195
4	0.8836	0.135
5	1.0487	0.091 (.)
6	1.2367	0.056 (.)
7	1.4219	0.034 (*)
8	1.6625	0.017 (*)

Table 1: Summary of logistic regression results for different values of N . Model formula: $\text{OCCURRENCE} \sim \text{DENSITY} + (1 + \text{DENSITY}|\text{VERB})$. Marginally significant effects are marked with a period (.), significant effects with a star (*).

For all values of N , we find a positive effect of DENSITY, i.e., there is a positive relation between the measure of density and the probability of first occurrence of a verb in the construction. However, the effect is only significant for $N \geq 7$; hence, the hypothesis that space density increases the odds of a coinage occurs in the construction is supported for measures of density based on these values of N .

More generally, the p -value decreases as N in-

creases, which means that the positive relation between DENSITY and OCCURRENCE is less systematic when DENSITY is measured with fewer neighbors. This is arguably because a higher N helps to better discriminate between dense clusters where all items are close together from looser ones that consist of a few ‘core’ items surrounded by more distant neighbors. This result illustrates the role of type frequency in syntactic productivity: a measure of density that is supported by a higher number of types makes better prediction than a measure supported by fewer types. This means that productivity not only hinges on how the existing semantic space relates to the novel item, it also occurs more reliably when this relation is attested by more items. These findings support the view that semantic density and type frequency, while they both positively influence syntactic productivity, do so in different ways: density defines the necessary conditions for a new coinage to occur, while type frequency increases the confidence that this coinage is indeed possible.

5 Conclusion

This paper reports the first attempt at using a distributional measure of semantic similarity derived from a vector-space model for the study of syntactic productivity in diachrony. On the basis of a case study of the construction “V *the hell out of NP*” from 1930 to 2009, the advantages of this approach were demonstrated. Not only does distributional semantics provide an empirically-based measure of semantic similarity that appropriately captures semantic distinctions, it also enables the use of methods for which quantification is necessary, such as data visualization and statistical analysis. Using multidimensional scaling and logistic regression, it was shown that the occurrence of new items throughout the history of the construction can be predicted by the density of the semantic space in the neighborhood of these items in prior usage. In conclusion, this work opens new perspectives for the study of syntactic productivity in line with the growing synergy between computational linguistics and other fields.

References

- Johana Barðdal. 2008. *Productivity: Evidence from Case and Argument Structure in Icelandic*. John Benjamins, Amsterdam.

- Douglas Bates, Martin Maechler, Ben Bolker and Steven Walker. 2011. *lme4: Linear mixed-effects models using S4 classes. R package*. URL: <http://CRAN.R-project.org/package=lme4>
- Joan Bybee. 2010. *Language, Usage and Cognition*. Cambridge University Press, Cambridge.
- Joan Bybee and David Eddington. 2006. A usage-based approach to Spanish verbs of ‘becoming’. *Language*, 82(2):323–355.
- Mark Davies. 2008. *The Corpus of Contemporary American English: 450 million words, 1990-present*. Available online at <http://corpus.byu.edu/coca/>
- Mark Davies. 2010. *The Corpus of Historical American English: 400 million words, 1810-2009*. Available online at <http://corpus.byu.edu/coha/>
- Georgiana Dinu, The Nghia Pham and Marco Baroni. 2013. DISSECT: DISTRIBUTIONAL SEMANTICS COMPOSITION TOOLKIT. In *Proceedings of the System Demonstrations of ACL 2013 (51st Annual Meeting of the Association for Computational Linguistics)*.
- Katrin Erk. 2012. Vector Space Models of Word Meaning and Phrase Meaning: A Survey. *Language and Linguistics Compass*, 6(10):635–653.
- Adele Goldberg. 1995. *Constructions: A construction grammar approach to argument structure*. University of Chicago Press, Chicago.
- Michael Israel. 1996. The way constructions grow. In Adele E. Goldberg (ed.), *Conceptual structure, discourse and language*, pages 217–230. CSLI Publications, Stanford, CA.
- Joseph Kruskal. 1964. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27.
- Alessandro Lenci. 2008. Distributional semantics in linguistic and cognitive research. *Rivista di Linguistica*, 20(1):1–31.
- George Miller and Walter Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28.
- R Development Core Team. 2013. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna; URL: <http://www.R-project.org/>
- Magnus Sahlgren. 2008. The distributional hypothesis. *Rivista di Linguistica*, 20(1):33–53.
- Helmut Schmid. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of International Conference on New Methods in Language Processing, Manchester, UK*.
- Laura Suttle and Adele Goldberg. 2011. The partial productivity of constructions as induction. *Linguistics*, 49(6):1237–1269.
- Peter Turney and Patrick Pantel. 2010. From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

Single Document Summarization based on Nested Tree Structure

Yuta Kikuchi[†] Tsutomu Hirao[‡] Hiroya Takamura[†] Manabu Okumura[†] Masaaki Nagata[‡]

[†]Tokyo Institute of technology

4295, Nagatsuta, Midori-ku, Yokohama, 226-8503, Japan

{kikuchi, takamura, oku}@lr.pi.titech.ac.jp

[‡]NTT Communication Science Laboratories, NTT Corporation

2-4, Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0237, Japan

{hirao.tsutomu, nagata.masaaki}@lab.ntt.co.jp

Abstract

Many methods of text summarization combining sentence selection and sentence compression have recently been proposed. Although the dependency between words has been used in most of these methods, the dependency between sentences, i.e., rhetorical structures, has not been exploited in such joint methods. We used both dependency between words and dependency between sentences by constructing a nested tree, in which nodes in the document tree representing dependency between sentences were replaced by a sentence tree representing dependency between words. We formulated a summarization task as a combinatorial optimization problem, in which the nested tree was trimmed without losing important content in the source document. The results from an empirical evaluation revealed that our method based on the trimming of the nested tree significantly improved the summarization of texts.

1 Introduction

Extractive summarization is one well-known approach to text summarization and extractive methods represent a document (or a set of documents) as a set of some textual units (e.g., sentences, clauses, and words) and select their subset as a summary. Formulating extractive summarization as a combinatorial optimization problem greatly improves the quality of summarization (McDonald, 2007; Filatova and Hatzivassiloglou, 2004; Takamura and Okumura, 2009). There has recently been increasing attention focused on approaches that jointly optimize sentence extraction and sentence compression (Tomita et al., 2009;

Qian and Liu, 2013; Morita et al., 2013; Gillick and Favre, 2009; Almeida and Martins, 2013; Berg-Kirkpatrick et al., 2011). We can only extract important content by trimming redundant parts from sentences.

However, as these methods did not include the discourse structures of documents, the generated summaries lacked coherence. It is important for generated summaries to have a discourse structure that is similar to that of the source document. Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) is one way of introducing the discourse structure of a document to a summarization task (Marcu, 1998; Daumé III and Marcu, 2002; Hirao et al., 2013). Hirao et al. recently transformed RST trees into dependency trees and used them for single document summarization (Hirao et al., 2013). They formulated the summarization problem as a tree knapsack problem with constraints represented by the dependency trees.

We propose a method of summarizing a single document that utilizes dependency between sentences obtained from rhetorical structures and dependency between words obtained from a dependency parser. We have explained our method with an example in Figure 1. First, we represent a document as a **nested tree**, which is composed of two types of tree structures: a **document tree** and a **sentence tree**. The document tree is a tree that has sentences as nodes and head modifier relationships between sentences obtained by RST as edges. The sentence tree is a tree that has words as nodes and head modifier relationships between words obtained by the dependency parser as edges. We can build the nested tree by regarding each node of the document tree as a sentence tree. Finally, we formulate the problem of single document summarization as that of combinatorial optimization, which is based on the trimming of the nested tree.

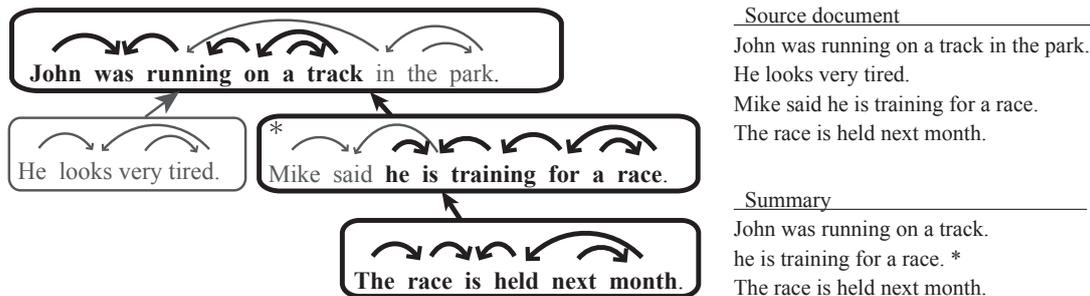


Figure 1: Overview of our method. The source document is represented as a nested tree. Our method simultaneously selects a rooted document subtree and sentence subtree from each node.

Our method jointly utilizes relations between sentences and relations between words, and extracts a rooted document subtree from a document tree whose nodes are arbitrary subtrees of the sentence tree.

Elementary Discourse Units (EDUs) in RST are defined as the minimal building blocks of discourse. EDUs roughly correspond to clauses. Most methods of summarization based on RST use EDUs as extraction textual units. We converted the rhetorical relations between EDUs to the relations between sentences to build the nested tree structure. We could thus take into account both relations between sentences and relations between words.

2 Related work

Extracting a subtree from the dependency tree of words is one approach to sentence compression (Tomita et al., 2009; Qian and Liu, 2013; Morita et al., 2013; Gillick and Favre, 2009). However, these studies have only extracted rooted subtrees from sentences. We allowed our model to extract a subtree that did not include the root word (See the sentence with an asterisk * in Figure 1). The method of Filippova and Strube (2008) allows the model to extract non-rooted subtrees in sentence compression tasks that compress a single sentence with a given compression ratio. However, it is not trivial to apply their method to text summarization because no compression ratio is given to sentences. None of these methods use the discourse structures of documents.

Daumé III and Marcu (2002) proposed a noisy-channel model that used RST. Although their method generated a well-organized summary, no optimality of information coverage was guaranteed and their method could not accept large texts because of the high computational cost. In addition,

-
- The scare over Alar, a growth regulator
 - that makes apples redder and crunchier
 - but may be carcinogenic,
 - made consumers shy away from the Delicious,
 - though they were less affected than the McIntosh.
-

Figure 2: Example of one sentence. Each line corresponds to one EDU.

tion, their method required large sets of data to calculate the accurate probability. There have been some studies that have used discourse structures locally to optimize the order of selected sentences (Nishikawa et al., 2010; Christensen et al., 2013).

3 Generating summary from nested tree

3.1 Building Nested Tree with RST

A document in RST is segmented into EDUs and adjacent EDUs are linked with rhetorical relations to build an RST-Discourse Tree (RST-DT) that has a hierarchical structure of the relations. There are 78 types of rhetorical relations between two spans, and each span has one of two aspects of a nucleus and a satellite. The nucleus is more salient to the discourse structure, while the other span, the satellite, represents supporting information. RST-DT is a tree whose terminal nodes correspond to EDUs and whose nonterminal nodes indicate the relations. Hirao et al. converted RST-DTs into dependency-based discourse trees (DEP-DTs) whose nodes corresponded to EDUs and whose edges corresponded to the head modifier relationships of EDUs. See Hirao et al. for details (Hirao et al., 2013).

Our model requires sentence-level dependency. Fortunately we can simply convert DEP-DTs to obtain dependency trees between sentences. We specifically merge EDUs that belong to the same sentence. Each sentence has only one root EDU that is the parent of all the other EDUs in the sentence. Each root EDU in a sentence has the parent

$$\begin{aligned}
\text{max.} \quad & \sum_i^n \sum_j^{m_i} w_{ij} z_{ij} \\
\text{s.t.} \quad & \sum_i^n \sum_j^{m_i} z_{ij} \leq L; & (1) \\
& x_{\text{parent}(i)} \geq x_i; & \forall i \quad (2) \\
& z_{\text{parent}(i,j)} - z_{ij} + r_{ij} \geq 0; & \forall i, j \quad (3) \\
& x_i \geq z_{ij}; & \forall i, j \quad (4) \\
& \sum_j^{m_i} z_{ij} \geq \min(\theta, \text{len}(i)) x_i; & \forall i \quad (5) \\
& \sum_j^{m_i} r_{ij} = x_i; & \forall i \quad (6) \\
& \sum_{j \notin R_c(i)} r_{ij} = 0; & \forall i \quad (7) \\
& r_{ij} \leq z_{ij}; & \forall i, j \quad (8) \\
& r_{ij} + z_{\text{parent}(i,j)} \leq 1; & \forall i, j \quad (9) \\
& r_{i\text{root}(i)} = z_{i\text{root}(i)}; & \forall i \quad (10) \\
& \sum_{j \in \text{sub}(i)} z_{ij} \geq x_i; & \forall i \quad (11) \\
& \sum_{j \in \text{obj}(i)} z_{ij} \geq x_i; & \forall i \quad (12)
\end{aligned}$$

Figure 3: ILP formulation ($x_i, z_{ij}, r_{ij} \in \{0, 1\}$)

EDU in another sentence. Hence, we can determine the parent-child relations between sentences. As a result, we obtain a tree that represents the parent-child relations of sentences, and we can use it as a document tree. After the document tree is obtained, we use a dependency parser to obtain the syntactic dependency trees of sentences. Finally, we obtain a nested tree.

3.2 ILP formulation

Our method generates a summary by trimming a nested tree. In particular, we extract a rooted document subtree from the document tree, and sentence subtrees from sentence trees in the document tree. We formulate our problem of optimization in this section as that of integer linear programming. Our model is shown in Figure 3.

Let us denote by w_{ij} the term weight of word ij (word j in sentence i). x_i is a variable that is one if sentence i is selected as part of a summary, and z_{ij} is a variable that is one if word ij is selected as part of a summary. According to the objective function, the score for the resulting summary is the sum of the term weights w_{ij} that are included in the summary. We denote by r_{ij} the variable that is one if word ij is selected as a root of an extracting sentence subtree. Constraint (1) guarantees that the summary length will be less than or equal to limit L . Constraints (2) and (3) are tree constraints for a document tree and sentence trees. r_{ij} in Constraint (3) allows the system

to extract non-rooted sentence subtrees, as we previously mentioned. Function $\text{parent}(i)$ returns the parent of sentence i and function $\text{parent}(i, j)$ returns the parent of word ij . Constraint (4) guarantees that words are only selected from a selected sentence. Constraint (5) guarantees that each selected sentence subtree has at least θ words. Function $\text{len}(i)$ returns the number of words in sentence i . Constraints (6)-(10) allow the model to extract subtrees that have an arbitrary root node. Constraint (6) guarantees that there is only one root per selected sentence. We can set the candidate for the root node of the subtree by using constraint (7). The $R_c(i)$ returns a set of the nodes that are the candidates of the root nodes in sentence i . It returned the parser’s root node and the verb nodes in this study. Constraint (8) maintains consistency between z_{ij} and r_{ij} . Constraint (9) prevents the system from selecting the parent node of the root node. Constraint (10) guarantees that the parser’s root node will only be selected when the system extracts a rooted sentence subtree. The $\text{root}(i)$ returns the word index of the parser’s root. Constraints (11) and (12) guarantee that the selected sentence subtree has at least one subject and one object if it has any. The $\text{sub}(i)$ and $\text{obj}(i)$ return the word indices whose dependency tag is “SUB” and “OBJ”.

3.3 Additional constraint for grammaticality

We added two types of constraints to our model to extract a grammatical sentence subtree from a dependency tree:

$$z_{ik} = z_{il}, \quad (13)$$

$$\sum_{k \in s(i,j)} z_{ik} = |s(i,j)| x_i. \quad (14)$$

Equation (13) means that words z_{ik} and z_{il} have to be selected together, i.e., a word whose dependency tag is PMOD or VC and its parent word, a negation and its parent word, a word whose dependency tag is SUB or OBJ and its parent verb, a comparative (JJR) or superlative (JJS) adjective and its parent word, an article (a/the) and its parent word, and the word “to” and its parent word. Equation (14) means that the sequence of words has to be selected together, i.e., a proper noun sequence whose POS tag is PRP\$, WP%, or POS and a possessive word and its parent word and the words between them. The $s(i, j)$ returns the set of word indices that are selected together with word ij .

Table 1: ROUGE score of each model. Note that the top two rows are both our proposals.

	ROUGE-1
Sentence subtree	0.354
Rooted sentence subtree	0.352
Sentence selection	0.254
EDU selection (Hirao et al., 2013)	0.321
LEAD _{EDU}	0.240
LEAD _{snt}	0.157

4 Experiment

4.1 Experimental Settings

We experimentally evaluated the test collection for single document summarization contained in the RST Discourse Treebank (RST-DTB) (Carlson et al., 2001) distributed by the Linguistic Data Consortium (LDC)¹. The RST-DTB Corpus includes 385 Wall Street Journal articles with RST annotations, and 30 of these documents also have one manually prepared reference summary. We set the length constraint, L , as the number of words in each reference summary. The average length of the reference summaries corresponded to approximately 10% of the length of the source document. This dataset was first used by Marcu et al. for evaluating a text summarization system (Marcu, 1998). We used ROUGE (Lin, 2004) as an evaluation criterion.

We compared our method (**sentence subtree**) with that of EDU selection (Hirao et al., 2013). We examined two other methods, i.e., **rooted sentence subtree** and **sentence selection**. These two are different from our method in the way that they select a sentence subtree. Rooted sentence subtree only selects rooted sentence subtrees². Sentence selection does not trim sentence trees. It simply selects full sentences from a document tree³. We built all document trees from the RST-DTs that were annotated in the corpus.

We set the term weight, w_{ij} , for our model as:

$$w_{ij} = \frac{\log(1 + tf_{ij})}{depth(i)^2}, \quad (15)$$

where tf_{ij} is the term frequency of word ij in a document and $depth(i)$ is the depth of sentence

¹<http://www ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2002T07>

²We achieved this by making $R_c(i)$ only return the parser’s root node in Figure 7.

³We achieved this by setting θ to a very large number.

i within the sentence-level DEP-DT that we described in Section 3.1. For Constraint (5), we set θ to eight.

4.2 Results and Discussion

4.2.1 Comparing ROUGE scores

We have summarized the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) scores for each method in Table 1. The score for sentence selection is low (0.254). However, introducing sentence compression to the system greatly improved the ROUGE score (0.354). The score is also higher than that with EDU selection, which is a state-of-the-art method. We applied a multiple test by using Holm’s method and found that our method significantly outperformed EDU selection and sentence selection. The difference between the sentence subtree and the rooted sentence subtree methods was fairly small. We therefore qualitatively analyzed some actual examples that will be discussed in Section 4.2.2. We also examined the ROUGE scores of two LEAD⁴ methods with different textual units: EDUs (LEAD_{EDU}) and sentences (LEAD_{SNT}). Although LEAD works well and often obtains high ROUGE scores for news articles, the scores for LEAD_{EDU} and LEAD_{SNT} were very low.

4.2.2 Qualitative Evaluation of Sentence Subtree Selection

This subsection compares the methods of subtree selection and rooted subtree selection. Figure 4 has two example sentences for which both methods selected a subtree as part of a summary. The $\{\cdot\}$ indicates the parser’s root word. The $[\cdot]$ indicates the word that the system selected as the root of the subtree. Subtree selection selected a root in both examples that differed from the parser’s root. As we can see, subtree selection only selected important subtrees that did not include the parser’s root, e.g., purpose-clauses and that-clauses. This capability is very effective because we have to contain important content in summaries within given length limits, especially when the compression ratio is high (i.e., the method has to generate much shorter summaries than the source documents).

⁴LEAD methods simply take the first K textual units from a source document until the summary length reaches L .

Original sentence	:	John Kriz, a Moody’s vice president, {said} Boston Safe Deposit’s performance has been hurt this year by a mismatch in the maturities of its assets and liabilities.
Rooted subtree selection	:	John Kriz a Moody’s vice president [{said}] Boston Safe Deposit’s performance has been hurt this year
Subtree selection	:	Boston Safe Deposit’s performance has [been] hurt this year
Original sentence	:	Recent surveys by Leo J. Shapiro & Associates, a market research firm in Chicago, {suggest} that Sears is having a tough time attracting shoppers because it hasn’t yet done enough to improve service or its selection of merchandise.
Rooted subtree selection	:	surveys [{suggest}] that Sears is having a time
Subtree selection	:	Sears [is] having a tough time attracting shoppers

Figure 4: Example sentences and subtrees selected by each method.

Table 2: Average number of words that individual extracted textual units contained.

Subtree	Sentence	EDU
15.29	18.96	9.98

4.2.3 Fragmentation of Information

Many studies that have utilized RST have simply adopted EDUs as textual units (Mann and Thompson, 1988; Daumé III and Marcu, 2002; Hirao et al., 2013; Knight and Marcu, 2000). While EDUs are textual units for RST, they are too fine grained as textual units for methods of extractive summarization. Therefore, the models have tended to select small fragments from many sentences to maximize objective functions and have led to fragmented summaries being generated. Figure 2 has an example of EDUs. A fragmented summary is generated when small fragments are selected from many sentences. Hence, the number of sentences in the source document included in the resulting summary can be an indicator to measure the fragmentation of information. We counted the number of sentences in the source document that each method used to generate a summary⁵. The average for our method was 4.73 and its median was four sentences. In contrast, methods of EDU selection had an average of 5.77 and a median of five sentences. This meant that our method generated a summary with a significantly smaller number of sentences⁶. In other words, our method relaxed fragmentation without decreasing the ROUGE score. There are boxplots of the numbers of selected sentences in Figure 5. Table 2 lists the number of words in each textual unit extracted by each method. It indicates that EDUs are shorter than the other textual units. Hence, the number of sentences tends to be large.

⁵Note that the number for the EDU method is not equal to selected textual units because a sentence in the source document may contain multiple EDUs.

⁶We used the Wilcoxon signed-rank test ($p < 0.05$).

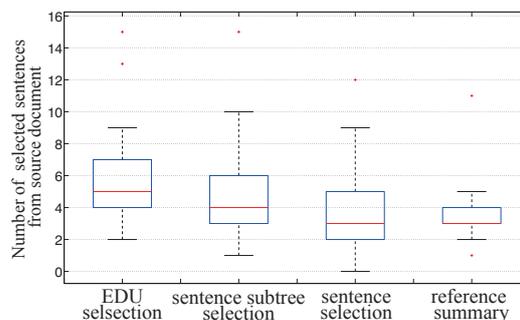


Figure 5: Number of sentences that each method selected.

5 Conclusion

We proposed a method of summarizing a single document that included relations between sentences and relations between words. We built a nested tree and formulated the problem of summarization as that of integer linear programming. Our method significantly improved the ROUGE score with significantly fewer sentences than the method of EDU selection. The results suggest that our method relaxed the fragmentation of information. We also discussed the effectiveness of sentence subtree selection that did not restrict rooted subtrees. Although ROUGE scores are widely used as evaluation metrics for text summarization systems, they cannot take into consideration linguistic qualities such as human readability. Hence, we plan to conduct evaluations with people⁷.

We only used the rhetorical structures between sentences in this study. However, there were also rhetorical structures between EDUs inside individual sentences. Hence, utilizing these for sentence compression has been left for future work. In addition, we used rhetorical structures that were manually annotated. There have been related studies on building RST parsers (duVerle and Prendinger, 2009; Hernault et al., 2010) and by using such parsers, we should be able to apply our model to other corpora or to multi-document settings.

⁷For example, the quality question metric from the Document Understanding Conference (DUC).

References

- Miguel Almeida and Andre Martins. 2013. Fast and robust compressive summarization with dual decomposition and multi-task learning. In *ACL*, pages 196–206, August.
- Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *ACL*, pages 481–490, Portland, Oregon, USA, June.
- Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2001. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *SIGDIAL*, pages 1–10.
- Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *NAACL:HLT*, pages 1163–1173.
- Hal Daumé III and Daniel Marcu. 2002. A noisy-channel model for document compression. *ACL*, pages 449–456.
- David duVerle and Helmut Prendinger. 2009. A novel discourse parser based on support vector machine classification. In *IJCNLP*, pages 665–673.
- Elena Filatova and Vasileios Hatzivassiloglou. 2004. A formal model for information selection in multi-sentence text extraction. In *COLING*.
- Katja Filippova and Michael Strube. 2008. Dependency tree based sentence compression. In *INLG*, pages 25–32.
- Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *ILP*, pages 10–18.
- Hugo Hernault, Helmut Prendinger, David duVerle, and Mitsuru Ishizuka. 2010. Hilda: A discourse parser using support vector machine classification. *Dialogue & Discourse*, 1(3):1–30.
- Tsutomu Hirao, Yasuhisa Yoshida, Masaaki Nishino, Norihito Yasuda, and Masaaki Nagata. 2013. Single-document summarization as a tree knapsack problem. In *EMNLP*, pages 1515–1520.
- Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization - step one: Sentence compression. In *National Conference on Artificial Intelligence (AAAI)*, pages 703–710.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out*, pages 74–81.
- William C. Mann and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, pages 243–281.
- Daniel Marcu. 1998. Improving summarization through rhetorical parsing tuning. In *In Proc. of the 6th Workshop on Very Large Corpora*, pages 206–215.
- Ryan T. McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *ECIR*, pages 557–564.
- Hajime Morita, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2013. Subtree extractive summarization via submodular maximization. In *ACL*, pages 1023–1032.
- Hitoshi Nishikawa, Takaaki Hasegawa, Yoshihiro Matsuo, and Genichiro Kikui. 2010. Opinion summarization with integer linear programming formulation for sentence extraction and ordering. In *COLING*, pages 910–918.
- Xian Qian and Yang Liu. 2013. Fast joint compression and summarization via graph cuts. In *EMNLP*, pages 1492–1502.
- Hiroya Takamura and Manabu Okumura. 2009. Text summarization model based on the budgeted median problem. In *CIKM*, pages 1589–1592.
- Kohei Tomita, Hiroya Takamura, and Manabu Okumura. 2009. A new approach of extractive summarization combining sentence selection and compression. *IPSJ SIG Notes*, pages 13–20.

Linguistic Considerations in Automatic Question Generation

Karen Mazidi

HiLT Lab
University of North Texas
Denton TX 76207, USA

KarenMazidi@my.unt.edu

Rodney D. Nielsen

HiLT Lab
University of North Texas
Denton TX 76207, USA

Rodney.Nielsen@unt.edu

Abstract

As students read expository text, comprehension is improved by pausing to answer questions that reinforce the material. We describe an automatic question generator that uses semantic pattern recognition to create questions of varying depth and type for self-study or tutoring. Throughout, we explore how linguistic considerations inform system design. In the described system, semantic role labels of source sentences are used in a domain-independent manner to generate both questions and answers related to the source sentence. Evaluation results show a 44% reduction in the error rate relative to the best prior systems, averaging over all metrics, and up to 61% reduction in the error rate on grammaticality judgments.

1 Introduction

Studies of student learning show that answering questions increases depth of student learning, facilitates transfer learning, and improves students' retention of material (McDaniel et al., 2007; Carpenter, 2012; Roediger and Pyc, 2012). The aim of this work is to automatically generate questions for such pedagogical purposes.

2 Related Work

Approaches to automatic question generation from text span nearly four decades. The vast majority of systems generate questions by selecting one sentence at a time, extracting portions of the source sentence, then applying transformation rules or patterns in order to construct a question. A well-known early work is Wolfe's AUTOQUEST (Wolfe, 1976), a syntactic pattern matching system. A recent approach from Heilman and Smith (2009, 2010) uses syntactic parsing and transformation rules to generate questions.

Syntactic, sentence-level approaches outnumber other approaches as seen in the Question Generation Shared Task Evaluation Challenge 2010 (Boyer and Piwek, 2010) which received only one paragraph-level, semantic entry. Argawal, Shah and Mannem (2011) continue the paragraph-level approach using discourse cues to find appropriate text segments upon which to construct questions at a deeper conceptual level. The uniqueness of their work lies in their use of discourse cues to extract semantic content for question generation. They generate questions of types: *why*, *when*, *give an example*, and *yes/no*.

In contrast to the above systems, other approaches have an intermediate step of transforming input into some sort of semantic representation. Examples of this intermediate step can be found in Yao and Zhang (2010) which uses Minimal Recursive Semantics, and in Olney et al. (2012) which uses concept maps. These approaches can potentially ask deeper questions due to their focus on semantics. A novel question generator by Curto et al. (2012) leverages lexico-syntactic patterns gleaned from the web with seed question-answer pairs.

Another recent approach is Lindberg et al. (2013), which used semantic role labeling to identify patterns in the source text from which questions can be generated. This work most closely parallels our own with a few exceptions: our system only asks questions that can be answered from the source text, our approach is domain-independent, and the patterns also identify the answer to the question.

3 Approach

The system consists of a straightforward pipeline. First, the source text is divided into sentences which are processed by SENNA¹ software, de-

¹<http://ml.nec-labs.com/senna/>

scribed in (Collobert et al., 2011). SENNA provides the tokenizing, pos tagging, syntactic constituency parsing and semantic role labeling used in the system. SENNA produces separate semantic role labels for each predicate in the sentence. For each predicate and its associated semantic arguments, a matcher function is called which will return a list of patterns that match the source sentence’s predicate-argument structure. Then questions are generated and stored by question type in a question hash table.

Generation patterns specify the text, verb forms and semantic arguments from the source sentence to form the question. Additionally, patterns indicate the semantic arguments that provide the answer to the question, required fields, and filter condition fields. As these patterns are matched, they will be rejected as candidates for generation for a particular sentence if the required arguments are absent or if filter conditions are present. For example, a filter for personal pronouns will prevent a question being generated with an argument that starts with a personal pronoun. From: *It means that the universe is expanding*, we do not want to generate a vague question such as: *What does it mean?* Coreference resolution, which could help avoid vague question generation, is discussed in Section 5. Table 1 shows selected required and filter fields, Section 3.3 gives examples of their use.

Patterns specify whether verbs should be included in their lexical form or as they appear in the source text. Either form will include subsequent particles such as: The lungs *take in* air. The most common use of the verb as it appears in the sentence is with the verb *be*, as in: What *were* fused into helium nuclei? This pattern takes the copular *be* as it appears in the source text. However, most patterns use the lexical form of the main verb along with the appropriate form of the auxiliary *do* (do, does, did), for the subject-auxiliary inversion required in forming interrogatives.

3.1 Pattern Authoring

The system at the time of this evaluation had 42 patterns. SENNA uses the 2005 PropBank coding scheme and we followed the documentation in (Babko-Malaya, 2005) for the patterns. The most commonly used semantic roles are A0, A1 and A2, as well as the ArgM modifiers.²

²Within PropBank, the precise roles of A0 - A6 vary by predicate.

Field	Meaning
Ax	Sentence must contain an Ax
!Ax	Sentence must not contain an Ax
AxPER	Ax must refer to a person
AxGER	Ax must contain a gerund
AxNN	Ax must contain nouns
!AxIN	Ax cannot start with a preposition
!AxPRP	Ax cannot start with per. pronoun
V= <i>verb</i>	Verb must be a form of <i>verb</i>
!be	Verb cannot be a form of <i>be</i>
negation	Sentence cannot contain negation

Table 1: Selected required and filter fields (*Ax is a semantic argument such as A0 or ArgM*)

3.2 Software Tools and Source Text

The system was created using SENNA and Python. Importing NLTK within Python provides a simple interface to WordNet from which we determine the lexical form of verbs. SENNA provided all the necessary processing of the data, quickly, accurately and in one run.

In order to generate questions, passages were selected from science textbooks downloaded from www.ck12.org. Textbooks were chosen rather than hand-crafted source material so that a more realistic assessment of performance could be achieved. For the experiments in this paper, we selected three passages from the subjects of biology, chemistry, and earth science, filtering out references to equations and figures. The passages average around 60 sentences each, and represent chapter sections. The average grade level is approximately grade 10 as indicated by the on-line readability scorer read-able.com.

3.3 Examples

Table 2 provides examples of generated questions. The pattern that generated Question 1 requires argument A1 (underlined in Table 2) and a causation ArgM (italicized). The pattern also filters out sentences with A0 or A2. The patterns are designed to match only the arguments used as part of the question or the answer, in order to prevent over generation of questions. The system inserted the correct forms of *release* and *do*, and ignored the phrase *As this occurs* since it is not part of the semantic argument.

The pattern that generated Question 2 requires A0, A1 and a verb whose lexical form is *mean* (V=*mean* in Table 1). In this pattern, A1 (itali-

<p>Question 1: Why did <u>potential energy</u> release? Answer: <i>because the new bonds have lower potential energy than the original bonds</i> Source: As this occurs, <u>potential energy</u> is released <i>because the new bonds have lower potential energy than the original bonds.</i></p>
<p>Question 2: What does <u>an increased surface area to volume ratio</u> indicate? Answer: <i>increased exposure to the environment</i> Source: <u>An increased surface area to volume ratio</u> means <i>increased exposure to the environment.</i></p>
<p>Question 3: What is another term for <u>electrically neutral particles</u>? Answer: <i>neutrons</i> Source: The nucleus contains positively charged particles called protons and <u>electrically neutral particles</u> called <i>neutrons.</i></p>
<p>Question 4: What happens if you continue to move atoms closer and closer together? Answer: <i>eventually the two nuclei will begin to repel each other</i> Source: <u>If you continue to move atoms closer and closer together,</u> <i>eventually the two nuclei will begin to repel each other.</i></p>

Table 2: Selected generated questions with source sentences

cized) forms the answer and A0 (underlined) becomes part of the question along with the appropriate form of *do*. This pattern supplies the word *indicate* instead of the source text's *mean* which broadens the question context.

Question 3 is from the source sentence's 3rd predicate-argument set because this matched the pattern requirements: A1, A2, V=call. The answer is the text from the A2 argument. The ability to generate questions from any predicate-argument set means that sentence simplification is not required as a preprocessing step, and that the sentence can match multiple patterns. For example, this sentence could also match patterns to generate questions such as: *What are positively charged particles called?* or *Describe the nucleus.*

Question 4 requires A1 and an ArgM that includes the discourse cue *if*. The ArgM (underlined) becomes part of the question, while the rest of the source sentence forms the answer. This pattern also requires that ArgM contain nouns (AxNN from Table 1), which helps filter vague questions.

4 Results

This paper focuses on evaluating generated questions primarily in terms of their linguistic quality, as did Heilman and Smith (2010a). In a related work (Mazidi and Nielsen, 2014) we evaluated the quality of the questions and answers from a pedagogical perspective, and our approach outperformed comparable systems in both linguistic and pedagogical evaluations. However, the task here is to explore the linguistic quality of generated

questions. The annotators are university students who are science majors and native speakers of English. Annotators were given instructions to read a paragraph, then the questions based on that paragraph. Two annotators evaluated each set of questions using Likert-scale ratings from 1 to 5, where 5 is the best rating, for grammaticality, clarity, and naturalness. The average inter-annotator agreement, allowing a difference of one between the annotators' ratings was 88% and Pearson's $r=0.47$ was statistically significant ($p<0.001$), suggesting a high correlation and agreement between annotators. The two annotator ratings were averaged for all the evaluations reported here.

We present results on three linguistic evaluations: (1) evaluation of our generated questions, (2) comparison of our generated questions with those from Heilman and Smith's question generator, and (3) comparison of our generated questions with those from Lindberg, Popowich, Nesbit and Winne. We compared our system to the H&S and LPN&W systems because they produce questions that are the most similar to ours, and for the same purpose: reading comprehension reinforcement. The Heilman and Smith system is available online;³ Lindberg graciously shared his code with us.

4.1 Evaluation of our Generated Questions

This evaluation was conducted with one file (Chemistry: Bonds) which had 59 sentences, from which the system generated 142 questions. The

³<http://www.ark.cs.cmu.edu/mheilman/questions/>

purpose of this evaluation was to determine if any patterns consistently produce poor questions. The average linguistics score per pattern in this evaluation was 5.0 to 4.18. We were also interested to know if first predicates make better questions than later ones. The average score by predicate position is shown in Table 3. Note that the Rating column gives the average of the grammaticality, clarity and naturalness scores.

Predicate	Questions	Rating
First	58	4.7
Second	35	4.7
Third	23	4.5
Higher	26	4.6

Table 3: Predicate depth and question quality

Based on this sample of questions there is no significant difference in linguistic scores for questions generated at various predicate positions. Some question generation systems simplify complex sentences in initial stages of their system. In our approach this is unnecessary, and simplifying could miss many valid questions.

4.2 Comparison with Heilman and Smith

This task utilized a file (Biology: the body) with 56 source sentences from which our system generated 102 questions. The Heilman and Smith system, as they describe it, takes an over-generate and rank approach. We only took questions that scored a 2.0 or better with their ranking system,⁴ which resulted in less than 27% of their top questions. In all, 84 of their questions were evaluated. The questions again were presented with accompanying paragraphs of the source text. Questions from the two systems were randomly intermingled. Annotators gave 1 - 5 scores for each category of grammaticality, clarity and naturalness.

As seen in Table 4, our results represent a 44% reduction in the error rate relative to Heilman and Smith on the average rating over all metrics, and as high as 61% reduction in the error rate on grammaticality judgments. The error reduction calculation is shown below. Note that *rating** is the maximum rating of 5.0.

$$\frac{rating_{system2} - rating_{system1}}{rating^* - rating_{system1}} \times 100.0 \quad (1)$$

⁴In our experiments, their rankings ranged from very small negative numbers to 3.0.

System	Gram	Clarity	Natural	Avg
H&S	4.38	4.13	3.94	4.15
M&N	4.76	4.26	4.53	4.52
Err. Red.	61%	15%	56%	44%

Table 4: Comparison with Heilman and Smith

System	Gram	Clarity	Natural	Avg
LPN&W	4.57	4.56	4.55	4.57
M&N	4.80	4.69	4.78	4.76
Err. Red.	54%	30%	51%	44%

Table 5: Comparison with Lindberg et al.

4.3 Comparison with Lindberg et al.

For a comparison with the Lindberg, Popowich, Nesbit and Winne system we used a file (Earth science: weather fronts) that seemed most similar to the text files for which their system was designed. The file has 93 sentences and our system generated 184 questions; the LPN&W system generated roughly 4 times as many questions. From each system, 100 questions were randomly selected, making sure that the LPN&W questions did not include questions generated from domain-specific templates such as: *Summarize the influence of the maximum amount on the environment.* The phrases *Summarize the influence of* and *on the environment* are part of a domain-specific template. The comparison results are shown in Table 5. Interestingly, our system again achieved a 44% reduction in the error rate when averaging over all metrics, just as it did in the Heilman and Smith comparison.

5 Linguistic Challenges

Natural language generation faces many linguistic challenges. Here we briefly describe three challenges: negation detection, coreference resolution, and verb forms.

5.1 Negation Detection

Negation detection is a complicated task because negation can occur at the word, phrase or clause level, and because there are subtle shades of negation between definite positive and negative polarities (Blanco and Moldovan, 2011). For our purposes we focused on negation as identified by the NEG label in SENNA which identified *not* in verb phrases. We have left for future work the task of

identifying other negative indicators, which occasionally does lead to poor question/answer quality as in the following:

Source sentence: In Darwin's time and today, many people incorrectly believe that evolution means humans come from monkeys.

Question: What does evolution mean?

Answer: that humans come from monkeys

The negation in the word *incorrectly* is not identified.

5.2 Coreference Resolution

Currently, our system does not use any type of coreference resolution. Experiments with existing coreference software performed well only for personal pronouns, which occur infrequently in most expository text. Not having coreference resolution leads to vague questions, some of which can be filtered as discussed previously. However, further work on filters is needed to avoid questions such as:

Source sentence: Air cools when it comes into contact with a cold surface or when it rises.

Question: What happens when it comes into contact with a cold surface or when it rises?

Heilman and Smith chose to filter out questions with personal pronouns, possessive pronouns and noun phrases composed simply of determiners such as *those*. Lindberg et al. used the emPronoun system from Charniak and Elsnier, which only handles personal pronouns. Since current state-of-the-art systems do not deal well with relative and possessive pronouns, this will continue to be a limitation of natural language generation systems for the time being.

5.3 Verb Forms

Since our focus is on expository text, system patterns deal primarily with the present and simple past tenses. Some patterns look for modals and so can handle future tense:

Source sentence: If you continue to move atoms closer and closer together, eventually the two nuclei will begin to repel each other.

Question: Discuss what the two nuclei will repel.

Light verbs pose complications in NLG because they are highly idiosyncratic and subject to syntactic variability (Sag et al., 2002). Light verbs can either carry semantic meaning (*take* your passport) or can be bleached of semantic content when

combined with other words as in: *make* a decision, *have* a drink, *take* a walk. Common English verbs that can be light verbs include give, have, make, take. Handling these constructions as well as other multi-word expressions may require both rule-based and statistical approaches. The catenative construction also potentially adds complexity (Huddleston and Pullum, 2005), as shown in this example: As the universe expanded, it became less dense and *began* to *cool*. Care must be taken not to generate questions based on one predicate in the catenative construction.

We are also hindered at times by the performance of the part of speech tagging and parsing software. The most common error observed was confusion between the noun and verb roles of a word. For example in: *Plant roots and bacterial decay use carbon dioxide in the process of respiration*, the word *use* was classified as NN, leaving no predicate and no semantic role labels in this sentence.

6 Conclusions

Roediger and Pyc (2012) advocate assisting students in building a strong knowledge base because creative discoveries are unlikely to occur when students do not have a sound set of facts and principles at their command. To that end, automatic question generation systems can facilitate the learning process by alternating passages of text with questions that reinforce the material learned.

We have demonstrated a semantic approach to automatic question generation that outperforms similar systems. We evaluated our system on text extracted from open domain STEM textbooks rather than hand-crafted text, showing the robustness of our approach. Our system achieved a 44% reduction in the error rate relative to both the Heilman and Smith, and the Lindberg et al. system on the average over all metrics. The results shows are statistically significant ($p < 0.001$). Our question generator can be used for self-study or tutoring, or by teachers to generate questions for classroom discussion or assessment. Finally, we addressed linguistic challenges to question generation.

Acknowledgments

This research was supported by the Institute of Education Sciences, U.S. Dept. of Ed., Grant R305A120808 to UNT. The opinions expressed are those of the authors.

References

- Agarwal, M., Shah, R., and Mannem, P. 2011. Automatic question generation using discourse cues. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, Association for Computational Linguistics.
- Babko-Malaya, O. 2005. Propbank annotation guidelines. URL: <http://verbs.colorado.edu>
- Blanco, E., and Moldovan, D. 2011. Some issues on detecting negation from text. In *FLAIRS Conference*.
- Boyer, K. E., and Piwek, P., editors. 2010. In *Proceedings of QG2010: The Third Workshop on Question Generation*. Pittsburgh: questiongeneration.org
- Carpenter, S. 2012. Testing enhances the transfer of learning. In *Current directions in psychological science*, 21(5), 279-283.
- Charniak, E., and Elsnar, M. 2009. EM works for pronoun anaphora resolution. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12, 2493-2537.
- Curto, S., Mendes, A., and Coheur, L. 2012. Question generation based on lexico-syntactic patterns learned from the web. *Dialogue & Discourse*, 3(2), 147-175.
- Heilman, M., and Smith, N. 2009. *Question generation via overgenerating transformations and ranking*. Technical Report CMU-LTI-09-013, Language Technologies Institute, Carnegie-Mellon University.
- Heilman, M., and Smith, N. 2010a. Good question! statistical ranking for question generation. In *Proceedings of NAACL/HLT 2010*. Association for Computational Linguistics.
- Heilman, M., and Smith, N. 2010b. Rating computer-generated questions with Mechanical Turk. In *Proceedings of the NAACL-HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. Association for Computational Linguistics.
- Huddleston, R. and Pullum, G. 2005. *A Student's Introduction to English Grammar*, Cambridge University Press.
- Lindberg, D., Popowich, F., Nesbit, J., and Winne, P. 2013. Generating natural language questions to support learning on-line. In *Proceedings of the 14th European Workshop on Natural Language Generation*, (2013): 105-114.
- Mannem, P., Prasad, R. and Joshi, A. 2010. Question generation from paragraphs at UPenn: QGSTEC system description. In *Proceedings of QG2010: The Third Workshop on Question Generation*.
- Mazidi, K. and Nielsen, R.D. 2014. Pedagogical evaluation of automatically generated questions. In *Intelligent Tutoring Systems*. LNCS 8474, Springer International Publishing Switzerland.
- McDaniel, M. A., Anderson, J. L., Derbish, M. H., and Morrisette, N. 2007. Testing the testing effect in the classroom. *European Journal of Cognitive Psychology*, 19(4-5), 494-513.
- Olney, A., Graesser, A., and Person, N. 2012. Question generation from concept maps. *Dialogue & Discourse*, 3(2), 75-99.
- Roediger III, H. L., and Pyc, M. 2012. Inexpensive techniques to improve education: Applying cognitive psychology to enhance educational practice. *Journal of Applied Research in Memory and Cognition*, 1.4: 242-248.
- Sag, I. A., Baldwin, T., Bond, F., Copestake, A., and Flickinger, D. 2002. Multiword expressions: A pain in the neck for NLP. In *Computational Linguistics and Intelligent Text Processing*, (pp. 1-15). Springer Berlin Heidelberg.
- Sternberg, R. J., & Grigorenko, E. L. 2003. Teaching for successful intelligence: Principles, procedures, and practices. *Journal for the Education of the Gifted*, 27, 207-228.
- Wolfe, J. 1976. Automatic question generation from text-an aid to independent study. In *Proceedings of ACM SIGCSE-SIGCUE*.
- Yao, X., and Zhang, Y. 2010. Question generation with minimal recursion semantics. In *Proceedings of QG2010: The Third Workshop on Question Generation*.

Polynomial Time Joint Structural Inference for Sentence Compression

Xian Qian and Yang Liu
The University of Texas at Dallas
800 W. Campbell Rd., Richardson, TX, USA
{qx, yangl}@hlt.utdallas.edu

Abstract

We propose two polynomial time inference algorithms to compress sentences under bigram and dependency-factored objectives. The first algorithm is exact and requires $O(n^6)$ running time. It extends Eisner’s cubic time parsing algorithm by using virtual dependency arcs to link deleted words. Two signatures are added to each span, indicating the number of deleted words and the rightmost kept word within the span. The second algorithm is a fast approximation of the first one. It relaxes the compression ratio constraint using Lagrangian relaxation, and thereby requires $O(n^4)$ running time. Experimental results on the popular sentence compression corpus demonstrate the effectiveness and efficiency of our proposed approach.

1 Introduction

Sentence compression aims to shorten a sentence by removing uninformative words to reduce reading time. It has been widely used in compressive summarization (Liu and Liu, 2009; Li et al., 2013; Martins and Smith, 2009; Chali and Hasan, 2012; Qian and Liu, 2013). To make the compressed sentence readable, some techniques consider the n-gram language models of the compressed sentence (Clarke and Lapata, 2008; McDonald, 2006). Recent studies used a subtree deletion model for compression (Berg-Kirkpatrick et al., 2011; Morita et al., 2013; Qian and Liu, 2013), which deletes a word only if its modifier in the parse tree is deleted. Despite its empirical success, such a model fails to generate compressions that are not subject to the subtree constraint (see Figure 1). In fact, we parsed the Edinburgh sentence compression corpus using the MSTparser¹,

¹<http://sourceforge.net/projects/mstparser/>

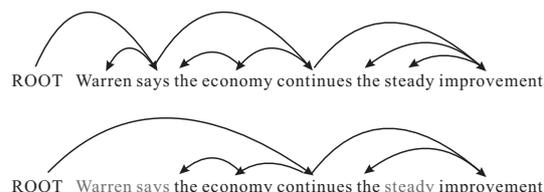


Figure 1: The compressed sentence is not a subtree of the original sentence. Words in gray are removed.

and found that 2561 of 5379 sentences (47.6%) do not satisfy the subtree deletion model.

Methods beyond the subtree model are also explored. Trevor et al. proposed synchronous tree substitution grammar (Cohn and Lapata, 2009), which allows local distortion of the tree topology and can thus naturally capture structural mismatches. (Genest and Lapalme, 2012; Thadani and McKeown, 2013) proposed the joint compression model, which simultaneously considers the n-gram model and dependency parse tree of the compressed sentence. However, the time complexity greatly increases since the parse tree dynamically depends on the compression. They used Integer Linear Programming (ILP) for inference which requires exponential running time in the worst case.

In this paper, we propose a new exact decoding algorithm for the joint model using dynamic programming. Our method extends Eisner’s cubic time parsing algorithm by adding signatures to each span, which indicate the number of deleted words and the rightmost kept word within the span, resulting in $O(n^6)$ time complexity and $O(n^4)$ space complexity. We further propose a faster approximate algorithm based on Lagrangian relaxation, which has $TO(n^4)$ running time and $O(n^3)$ space complexity (T is the iteration number in the subgradient decent algorithm). Experiments on the popular Edinburgh dataset show that

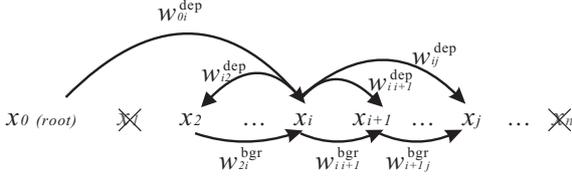


Figure 2: Graph illustration for the objective function. In this example, words x_2, x_i, x_{i+1}, x_j are kept, others are deleted. The value of the objective function is $w_2^{\text{tok}} + w_i^{\text{tok}} + w_{i+1}^{\text{tok}} + w_j^{\text{tok}} + w_{0i}^{\text{dep}} + w_{i2}^{\text{dep}} + w_{ii+1}^{\text{dep}} + w_{ij}^{\text{dep}} + w_{2i}^{\text{bgr}} + w_{ii+1}^{\text{bgr}} + w_{i+1j}^{\text{bgr}}$.

the proposed approach is 10 times faster than a high-performance commercial ILP solver.

2 Task Definition

We define the sentence compression task as: given a sentence composed of n words, $\mathbf{x} = x_1, \dots, x_n$, and a length $L \leq n$, we need to remove $(n - L)$ words from \mathbf{x} , so that the sum of the weights of the dependency tree and word bigrams of the remaining part is maximized. Formally, we solve the following optimization problem:

$$\begin{aligned} \max_{\mathbf{z}, \mathbf{y}} \quad & \sum_i w_i^{\text{tok}} z_i + \sum_{i,j} w_{ij}^{\text{dep}} z_i z_j y_{ij} \quad (1) \\ & + \sum_{i < j} w_{ij}^{\text{bgr}} z_i z_j \prod_{i < k < j} (1 - z_k) \\ \text{s.t.} \quad & \mathbf{z} \text{ is binary, } \sum_i z_i = L \\ & \mathbf{y} \text{ is a projective parse tree over the} \\ & \text{subgraph: } \{x_i | z_i = 1\} \end{aligned}$$

where \mathbf{z} is a binary vector, z_i indicates x_i is kept or not. \mathbf{y} is a square matrix denoting the projective dependency parse tree over the remaining words, y_{ij} indicates if x_i is the head of x_j (note that each word has exactly one head). w_i^{tok} is the informativeness of x_i , w_{ij}^{bgr} is the score of bigram $x_i x_j$ in an n -gram model, w_{ij}^{dep} is the score of dependency arc $x_i \rightarrow x_j$ in an arc-factored dependency parsing model. Hence, the first part of the objective function is the total score of the kept words, the second and third parts are the scores of the parse tree and bigrams of the compressed sentence, $z_i z_j \prod_{i < k < j} (1 - z_k) = 1$ indicates both x_i and x_j are kept, and are adjacent after compression. A graph illustration of the objective function is shown in Figure 2.



Figure 3: Connect deleted words using virtual arcs.

3 Proposed Method

3.1 Eisner's Cubic Time Parsing Algorithm

Throughout the paper, we assume that all the parse trees are projective. Our method is a generalization of Eisner's dynamic programming algorithm (Eisner, 1996), where two types of structures are used in each iteration, incomplete spans and complete spans. A span is a subtree over a number of consecutive words, with the leftmost or the rightmost word as its root. An incomplete span denoted as I_j^i is a subtree inside a single arc $x_i \rightarrow x_j$, with root x_i . A complete span is denoted as C_j^i , where x_i is the root of the subtree, and x_j is the furthest descendant of x_i .

Eisner's algorithm searches the optimal tree in a bottom up order. In each step, it merges two adjacent spans into a larger one. There are two rules for merging spans: one merges two complete spans into an incomplete span, the other merges an incomplete span and a complete span into a large complete span.

3.2 Exact $O(n^6)$ Time Algorithm

First we consider an easy case, where the bigram scores w_{ij}^{bgr} in the objective function are ignored.

The scores of unigrams w_i^{tok} can be transferred to the dependency arcs, so that we can remove all linear terms $w_i^{\text{tok}} z_i$ from the objective function. That is:

$$\begin{aligned} & \sum_i w_i^{\text{tok}} z_i + \sum_{i,j} w_{ij}^{\text{dep}} z_i z_j y_{ij} \\ & = \sum_{i,j} (w_{ij}^{\text{dep}} + w_j^{\text{tok}}) z_i z_j y_{ij} \end{aligned}$$

This can be easily verified. If $z_j = 0$, then in both equations, all terms having z_j are zero; If $z_j = 1$, i.e., x_j is kept, since it has exactly one head word x_k in the compressed sentence, the sum of the terms having z_j is $w_j^{\text{tok}} + w_{kj}^{\text{dep}}$ for both equations.

Therefore, we only need to consider the scores of arcs. For any compressed sentence, we could augment its dependency tree by adding a virtual

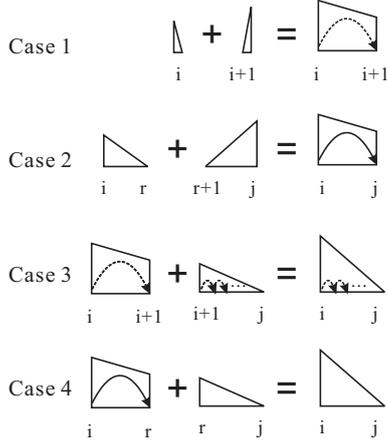


Figure 4: Merging rules for dependency-factored sentence compression. Incomplete spans and complete spans are represented by trapezoids and triangles respectively.

arc $i - 1 \rightarrow i$ for each deleted word x_i . If the first word x_1 is deleted, we connect it to the root of the parse tree x_0 , as shown in Figure 3. In this way, we derive a full parse tree of the original sentence. This is a one-to-one mapping. We can reversely get the the compressed parse tree by removing all virtual arcs from the full parse tree. We restrict the score of all the virtual arcs to be zero, so that scores of the two parse trees are equivalent.

Now the problem is to search the optimal full parse tree with $n - L$ virtual arcs.

We modify Eisner’s algorithm by adding a signature to each span indicating the number of virtual arcs within the span. Let $I_j^i(k)$ and $C_j^i(k)$ denote the incomplete and complete spans with k virtual arcs respectively. When merging two spans, there are 4 cases, as shown in Figure 4.

- **Case 1** Link two complete spans by a virtual arc : $I_{i+1}^i(1) = C_i^i(0) + C_{i+1}^{i+1}(0)$.

The two complete spans must be single words, as the length of the virtual arc is 1.

- **Case 2** Link two complete spans by a non-virtual arc: $I_j^i(k) = C_r^i(k') + C_{r+1}^j(k''), k' + k'' = k$.

- **Case 3** Merge an incomplete span and a complete span. The incomplete span is covered by a virtual arc: $I_j^i(j - i) = I_{i+1}^i(1) + C_j^{i+1}(j - i - 1)$. The number of the virtual arcs within C_j^{i+1} must be $j - i - 1$, since

the descendants of the modifier of a virtual arc x_j must be removed.

- **Case 4** Merge an incomplete span and a complete span. The incomplete span is covered by a non-virtual arc: $C_j^i(k) = I_r^i(k') + C_j^r(k''), k' + k'' = k$.

The score of the new span is the sum of the two spans. For case 2, the weight of the dependency arc $i \rightarrow j$, w_{ij}^{dep} is also added to the final score. The root node is allowed to have two modifiers: one is the modifier in the compressed sentence, the other is the first word if it is removed.

For each combination, the algorithm enumerates the number of virtual arcs in the left and right spans, and the split position (e.g., k', k'', r in case 2), thus it takes $O(n^3)$ running time. The overall time complexity is $O(n^5)$ and the space complexity is $O(n^3)$.

Next, we consider the bigram scores. The following proposition is obvious.

Proposition 1. For any right-headed span I_j^i or C_j^i , $i > j$, words x_i, x_j must be kept.

Proof. Suppose x_j is removed, there must be a virtual arc $j - 1 \rightarrow j$ which is a conflict with the fact that x_j is the leftmost word. As x_j is a descendant of x_i , x_i must be kept. \square

When merging two spans, a new bigram is created, which connects the rightmost kept words in the left span and the leftmost kept word in the right span. According to the proposition above, if the right span is right-headed, its leftmost word is kept. If the right span is left-headed, there are two cases: its leftmost word is kept, or no word in the span is kept. In any case, we only need to consider the leftmost word in the right span.

Let $I_j^i(k, p)$ and $C_j^i(k, p)$ denote the single and complete span with k virtual arcs and the rightmost kept word x_p . According to the proposition above, we have, for any right-headed span $p = i$.

We slightly modify the two merging rules above, and obtain:

- **Case 2'** Link two complete spans by a non-virtual arc: $I_j^i(k, j) = C_r^i(k', p) + C_{r+1}^j(k'', j), k' + k'' = k$. The score of the new span is the sum of the two spans plus $w_{ij}^{\text{dep}} + w_{p, r+1}^{\text{bgr}}$.

- **Case 4'** Merge an incomplete span and a complete span. The incomplete span is covered by a non-virtual arc. For left-headed spans, the rule is $C_j^i(k, q) = I_r^i(k', p) + C_j^r(k'', q)$, $k' + k'' = k$, and the score of the new span is the sum of the two spans plus w_{pr}^{bgr} ; for right-headed spans, the rule is $C_j^i(k, i) = I_r^i(k', i) + C_j^r(k'', r)$, and the score of the new span is the sum of the two spans.

The modified algorithm requires $O(n^6)$ running time and $O(n^4)$ space complexity.

3.3 Approximate $O(n^4)$ Time Algorithm

In this section, we propose an approximate algorithm where the length constraint $\sum_i z_i = L$ is relaxed by Lagrangian Relaxation. The relaxed version of Problem (1) is

$$\begin{aligned} \min_{\lambda} \max_{\mathbf{z}, \mathbf{y}} \quad & \sum_i w_i^{\text{tok}} z_i + \sum_{i,j} w_{ij}^{\text{dep}} z_i z_j y_{ij} \quad (2) \\ & + \sum_{i < j} w_{ij}^{\text{bgr}} z_i z_j \prod_{i < k < j} (1 - z_k) \\ & + \lambda (\sum_i z_i - L) \\ \text{s.t.} \quad & \mathbf{z} \text{ is binary} \\ & \mathbf{y} \text{ is a projective parse tree over the} \\ & \text{subgraph: } \{x_i | z_i = 1\} \end{aligned}$$

Fixing λ , the optimal \mathbf{z}, \mathbf{y} can be found using a simpler version of the algorithm above. We drop the signature of the virtual arc number from each span, and thus obtain an $O(n^4)$ time algorithm. Space complexity is $O(n^3)$. Fixing \mathbf{z}, \mathbf{y} , the dual variable is updated by

$$\lambda = \lambda + \alpha (L - \sum_i z_i)$$

where $\alpha > 0$ is the learning rate. In this paper, our choice of α is the same as (Rush et al., 2010).

4 Experiments

4.1 Data and Settings

We evaluate our method on the data set from (Clarke and Lapata, 2008). It includes 82 newswire articles with manually produced compression for each sentence. We use the same partitions as (Martins and Smith, 2009), i.e., 1,188 sentences for training and 441 for testing.

Our model is discriminative – the scores of the unigrams, bigrams and dependency arcs are the linear functions of features, that is, $w_i^{\text{tok}} = \mathbf{v}^T \mathbf{f}(x_i)$, where \mathbf{f} is the feature vector of x_i , and \mathbf{v} is the weight vector of features. The learning task is to estimate the feature weight vector based on the manually compressed sentences.

We run a second order dependency parser trained on the English Penn Treebank corpus to generate the parse trees of the compressed sentences. Then we augment these parse trees by adding virtual arcs and get the full parse trees of their corresponding original sentences. In this way, the annotation is transformed into a set of sentences with their augmented parse trees. The learning task is similar to training a parser. We run a CRF based POS tagger to generate POS related features.

We adopt the compression evaluation metric as used in (Martins and Smith, 2009) that measures the macro F-measure for the retained unigrams (F_{ugr}), and the one used in (Clarke and Lapata, 2008) that calculates the F1 score of the grammatical relations labeled by RASP (Briscoe and Carroll, 2002).

We compare our method with other 4 state-of-the-art systems. The first is linear chain CRFs, where the compression task is casted as a binary sequence labeling problem. It usually achieves high unigram F1 score but low grammatical relation F1 score since it only considers the local interdependence between adjacent words. The second is the subtree deletion model (Berg-Kirkpatrick et al., 2011) which is solved by integer linear programming (ILP)². The third one is the bigram model proposed by McDonald (McDonald, 2006) which adopts dynamic programming for efficient inference. The last one jointly infers tree structures alongside bigrams using ILP (Thadani and McKeown, 2013). For fair comparison, systems were restricted to produce compressions that matched their average gold compression rate if possible.

4.2 Features

Three types of features are used to learn our model: unigram features, bigram features and dependency features, as shown in Table 1. We also use the in-between features proposed by (McDonald et

²We use Gurobi as the ILP solver in the paper. <http://www.gurobi.com/>

Features for unigram x_i
$w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}$ $t_{i-2}, t_{i-1}, t_i, t_{i+1}, t_{i+2}$ $w_i t_i$ $w_{i-1} w_i, w_i w_{i+1}$ $t_{i-2} t_{i-1}, t_{i-1} t_i, t_i t_{i+1}, t_{i+1} t_{i+2}$ $t_{i-2} t_{i-1} t_i, t_{i-1} t_i t_{i+1}, t_i t_{i+1} t_{i+2}$ whether w_i is a stopword
Features for selected bigram $x_i x_j$
distance between the two words: $j - i$ $w_i w_j, w_{i-1} w_j, w_{i+1} w_j, w_i w_{j-1}, w_i w_{j+1}$ $t_i t_j, t_{i-1} t_j, t_{i+1} t_j, t_i t_{j-1}, t_i t_{j+1}$ Concatenation of the templates above $\{t_i t_k t_j i < k < j\}$
Dependency Features for arc $x_h \rightarrow x_m$
distance between the head and modifier $h - m$ dependency type direction of the dependency arc (left/right) $w_h w_m, w_{h-1} w_m, w_{h+1} w_m, w_h w_{m-1}, w_h w_{m+1}$ $t_h t_m, t_{h-1} t_m, t_{h+1} t_m, t_h t_{m-1}, t_h t_{m+1}$ $t_{h-1} t_h t_{m-1} t_m, t_h t_{h+1} t_{m-1} t_m$ $t_{h-1} t_h t_m t_{m+1}, t_h t_{h+1} t_m t_{m+1}$ Concatenation of the templates above $\{t_h t_k t_m x_k \text{ lies between } x_h \text{ and } x_m\}$

Table 1: Feature templates. w_i denotes the word form of token x_i and t_i denotes the POS tag of x_i .

al., 2005), which were shown to be very effective for dependency parsing.

4.3 Results

We show the comparison results in Table 2. As expected, the joint models (ours and TM13) consistently outperform the subtree deletion model, since the joint models do not suffer from the subtree restriction. They also outperform McDonald’s, demonstrating the effectiveness of considering the grammar structure for compression. It is not surprising that CRFs achieve high unigram F scores but low syntactic F scores as they do not

System	C Rate	F_{uni}	RASP	Sec.
Ours(Approx)	0.68	0.802	0.598	0.056
Ours(Exact)	0.68	0.805	0.599	0.610
Subtree	0.68	0.761	0.575	0.022
TM13	0.68	0.804	0.599	0.592
McDonald06	0.71	0.776	0.561	0.010
CRFs	0.73	0.790	0.501	0.002

Table 2: Comparison results under various quality metrics, including unigram F1 score (F_{uni}), syntactic F1 score (RASP), and compression speed (seconds per sentence). C Rate is the compression ratio of the system generated output. For fair comparison, systems were restricted to produce compressions that matched their average gold compression rate if possible.

consider the fluency of the compressed sentence.

Compared with TM13’s system, our model with exact decoding is not significantly faster due to the high order of the time complexity. On the other hand, our approximate approach is much more efficient, about 10 times faster than TM13’s system, and achieves competitive accuracy with the exact approach. Note that it is worth pointing out that the exact approach can output compressed sentences of all lengths, whereas the approximate method can only output one sentence at a specific compression rate.

5 Conclusion

In this paper, we proposed two polynomial time decoding algorithms using joint inference for sentence compression. The first one is an exact dynamic programming algorithm, and requires $O(n^6)$ running time. This one does not show significant advantage in speed over ILP. The second one is an approximation of the first algorithm. It adopts Lagrangian relaxation to eliminate the compression ratio constraint, yielding lower time complexity $TO(n^4)$. In practice it achieves nearly the same accuracy as the exact one, but is much faster.³

The main assumption of our method is that the dependency parse tree is projective, which is not true for some other languages. In that case, our method is invalid, but (Thadani and McKeown, 2013) still works. In the future, we will study the non-projective cases based on the recent parsing techniques for 1-endpoint-crossing trees (Pitler et al., 2013).

Acknowledgments

We thank three anonymous reviewers for their valuable comments. This work is partly supported by NSF award IIS-0845484 and DARPA under Contract No. FA8750-13-2-0041. Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

References

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of ACL-HLT*, pages 481–490, June.

³Our code is available at <http://code.google.com/p/sent-compress/>

- T. Briscoe and J. Carroll. 2002. Robust accurate statistical annotation of general text.
- Yllias Chali and Sadid A. Hasan. 2012. On the effectiveness of using sentence compression models for query-focused multi-document summarization. In *Proceedings of COLING*, pages 457–474.
- James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *J. Artif. Intell. Res. (JAIR)*, 31:399–429.
- Trevor Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *J. Artif. Int. Res.*, 34(1):637–674, April.
- Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: an exploration. In *Proceedings of COLING*.
- Pierre-Etienne Genest and Guy Lapalme. 2012. Fully abstractive approach to guided summarization. In *Proceedings of the ACL*, pages 354–358.
- Chen Li, Fei Liu, Fuliang Weng, and Yang Liu. 2013. Document summarization via guided sentence compression. In *Proceedings of EMNLP*, October.
- Fei Liu and Yang Liu. 2009. From extractive to abstractive meeting summaries: Can it be done by sentence compression? In *Proceedings of ACL-IJCNLP 2009*, pages 261–264, August.
- André F. T. Martins and Noah A. Smith. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 1–9.
- Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL*.
- Ryan McDonald. 2006. Discriminative Sentence Compression with Soft Syntactic Constraints. In *Proceedings of EACL*, April.
- Hajime Morita, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2013. Subtree extractive summarization via submodular maximization. In *Proceedings of ACL*, pages 1023–1032, August.
- Emily Pitler, Sampath Kannan, and Mitchell Marcus. 2013. Finding optimal 1-endpoint-crossing trees. In *Transactions of the Association for Computational Linguistics, 2013 Volume 1*.
- Xian Qian and Yang Liu. 2013. Fast joint compression and summarization via graph cuts. In *Proceedings of EMNLP*, pages 1492–1502, October.
- Alexander M Rush, David Sontag, Michael Collins, and Tommi Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of EMNLP*.
- Kapil Thadani and Kathleen McKeown. 2013. Sentence compression with joint structural inference. In *Proceedings of the CoNLL*, August.

A Bayesian Method to Incorporate Background Knowledge during Automatic Text Summarization

Annie Louis

ILCC, School of Informatics,
University of Edinburgh,
Edinburgh EH8 9AB, UK
alouis@inf.ed.ac.uk

Abstract

In order to summarize a document, it is often useful to have a *background* set of documents from the domain to serve as a reference for determining new and important information in the input document. We present a model based on Bayesian surprise which provides an intuitive way to identify surprising information from a summarization input with respect to a background corpus. Specifically, the method quantifies the degree to which pieces of information in the input change one's beliefs' about the world represented in the background. We develop systems for generic and update summarization based on this idea. Our method provides competitive content selection performance with particular advantages in the update task where systems are given a small and topical background corpus.

1 Introduction

Important facts in a new text are those which deviate from previous knowledge on the topic. When people create summaries, they use their knowledge about the world to decide what content in an input document is informative to include in a summary. Understandably in automatic summarization as well, it is useful to keep a background set of documents to represent general facts and their frequency in the domain.

For example, in the simplest setting of multi-document summarization of news, systems are asked to summarize an *input set* of topically-related news documents to reflect its central content. In this *GENERIC* task, some of the best reported results were obtained by a system (Conroy et al., 2006) which computed importance scores for words in the input by examining if the word

occurs with significantly higher probability in the input compared to a large background collection of news articles. Other specialized summarization tasks explicitly require the use of background information. In the *UPDATE* summarization task, a system is given two sets of news documents on the same topic; the second contains articles published later in time. The system should summarize the important updates from the second set assuming a user has already read the first set of articles.

In this work, we present a Bayesian model for assessing the novelty of a sentence taken from a summarization input with respect to a background corpus of documents.

Our model is based on the idea of Bayesian Surprise (Itti and Baldi, 2006). For illustration, assume that a user's background knowledge comprises of multiple hypotheses about the current state of the world and a probability distribution over these hypotheses indicates his degree of belief in each hypothesis. For example, one hypothesis may be that *the political situation in Ukraine is peaceful*, another where *it is not*. Apriori assume the user favors the hypothesis about a peaceful Ukraine, i.e. the hypothesis has higher probability in the prior distribution. Given new data, the evidence can be incorporated using Bayes Rule to compute the posterior distribution over the hypotheses. For example, upon viewing news reports about riots in the country, a user would update his beliefs and the posterior distribution of the user's knowledge would have a higher probability for a riotous Ukraine. Bayesian surprise is the difference between the prior and posterior distributions over the hypotheses which quantifies the extent to which the new data (the news report) has changed a user's prior beliefs about the world.

In this work, we exemplify how Bayesian surprise can be used to do content selection for text summarization. Here a user's prior knowledge is approximated by a background corpus and we

show how to identify sentences from the input set which are most surprising with respect to this background. We use the method to do two types of summarization tasks: a) GENERIC news summarization which uses a large random collection of news articles as the background, and b) UPDATE summarization where the background is a smaller but specific set of news documents on the same topic as the input set. We find that our method performs competitively with a previous log-likelihood ratio approach which identifies words with significantly higher probability in the input compared to the background. The Bayesian approach is more advantageous in the update task, where the background corpus is smaller in size.

2 Related work

Computing new information is useful in many applications. The TREC novelty tasks (Allan et al., 2003; Soboroff and Harman, 2005; Schiffman, 2005) tested the ability of systems to find novel information in an IR setting. Systems were given a list of documents ranked according to relevance to a query. The goal is to find sentences in each document which are relevant to the query, and at the same time is new information given the content of documents higher in the relevance list.

For update summarization of news, methods range from textual entailment techniques (Bentivogli et al., 2010) to find facts in the input which are not entailed by the background, to Bayesian topic models (Delort and Alfonseca, 2012) which aim to learn and use topics discussed only in background, those only in the update input and those that overlap across the two sets.

Even for generic summarization, some of the best results were obtained by Conroy et al. (2006) by using a large random corpus of news articles as the background while summarizing a new article, an idea first proposed by Lin and Hovy (2000). Central to this approach is the use of a likelihood ratio test to compute *topic words*, words that have significantly higher probability in the input compared to the background corpus, and are hence descriptive of the input’s topic. In this work, we compare our system to topic word based ones since the latter is also a general method to find surprising new words in a set of input documents but is not a bayesian approach. We briefly explain the topic words based approach below.

Computing topic words: Let us call the input

set I and the background B . The log-likelihood ratio test compares two hypotheses:

H_1 : A word t is not a topic word and occurs with equal probability in I and B , i.e. $p(t|I) = p(t|B) = p$

H_2 : t is a topic word, hence $p(t|I) = p_1$ and $p(t|B) = p_2$ and $p_1 > p_2$

A set of documents D containing N tokens is viewed as a sequence of words $w_1 w_2 \dots w_N$. The word in each position i is assumed to be generated by a Bernoulli trial which succeeds when the generated word $w_i = t$ and fails when w_i is not t . Suppose that the probability of success is p . Then the probability of a word t appearing k times in a dataset of N tokens is the binomial probability:

$$b(k, N, p) = \binom{N}{k} p^k (1-p)^{N-k} \quad (1)$$

The likelihood ratio compares the likelihood of the data $D = \{B, I\}$ under the two hypotheses.

$$\lambda = \frac{P(D|H_1)}{P(D|H_2)} = \frac{b(c_t, N, p)}{b(c_I, N_I, p_1) b(c_B, N_B, p_2)} \quad (2)$$

p , p_1 and p_2 are estimated by maximum likelihood. $p = c_t/N$ where c_t is the number of times word t appears in the total set of tokens comprising $\{B, I\}$. $p_1 = c_t^I/N_I$ and $p_2 = c_t^B/N_B$ are the probabilities of t estimated only from the input and only from the background respectively.

A convenient aspect of this approach is that $-2 \log \lambda$ is asymptotically χ^2 distributed. So for a resulting $-2 \log \lambda$ value, we can use the χ^2 table to find the significance level with which the null hypothesis H_1 can be rejected. For example, a value of 10 corresponds to a significance level of 0.001 and is standardly used as the cutoff. Words with $-2 \log \lambda > 10$ are considered topic words. Conroy et al. (2006)’s system gives a weight of 1 to the topic words and scores sentences using the number of topic words normalized by sentence length.

3 Bayesian Surprise

First we present the formal definition of Bayesian surprise given by Itti and Baldi (2006) without reference to the summarization task.

Let \mathbf{H} be the space of all hypotheses representing the background knowledge of a user. The user has a probability $P(H)$ associated with each hypothesis $H \in \mathbf{H}$. Let D be a new observation. The posterior probability of a single hypothesis H can be computed as:

$$P(H|D) = \frac{P(D|H)P(H)}{P(D)} \quad (3)$$

The surprise $S(D, \mathbf{H})$ created by D on hypothesis space \mathbf{H} is defined as the difference between the prior and posterior distributions over the hypotheses, and is computed using KL divergence.

$$S(D, \mathbf{H}) = \text{KL}(P(H|D), P(H)) \quad (4)$$

$$= \int_{\mathbf{H}} P(H|D) \log \frac{P(H|D)}{P(H)} \quad (5)$$

Note that since KL-divergence is not symmetric, we could also compute $\text{KL}(P(H), P(H|D))$ as the surprise value. In some cases, surprise can be computed analytically, in particular when the prior distribution is conjugate to the form of the hypothesis, and so the posterior has the same functional form as the prior. (See Baldi and Itti (2010) for the surprise computation for different families of probability distributions).

4 Summarization with Bayesian Surprise

We consider the hypothesis space \mathbf{H} as the set of all the hypotheses encoding background knowledge. A single hypothesis about the background takes the form of a multinomial distribution over word unigrams. For example, one multinomial may have higher word probabilities for ‘Ukraine’ and ‘peaceful’ and another multinomial has higher probabilities for ‘Ukraine’ and ‘riots’. $P(H)$ gives a prior probability to each hypothesis based on the information in the background corpus. In our case, $P(H)$ is a Dirichlet distribution, the conjugate prior for multinomials. Suppose that the vocabulary size of the background corpus is V and we label the word types as (w_1, w_2, \dots, w_V) . Then,

$$P(H) = \text{Dir}(\alpha_1, \alpha_2, \dots, \alpha_V) \quad (6)$$

where $\alpha_{1:V}$ are the concentration parameters of the Dirichlet distribution (and will be set using the background corpus as explained in Section 4.2).

Now consider a new observation I (a text, sentence, or paragraph from the *summarization input*) and the word counts in I given by (c_1, c_2, \dots, c_V) . Then the posterior over H is the dirichlet:

$$P(H|I) = \text{Dir}(\alpha_1 + c_1, \alpha_2 + c_2, \dots, \alpha_V + c_V) \quad (7)$$

The surprise due to observing I , $S(I, \mathbf{H})$ is the KL divergence between the two dirichlet distributions. (Details about computing KL divergence between two dirichlet distributions can be found in Penny (2001) and Baldi and Itti (2010)).

Below we propose a general algorithm for summarization using surprise computation. Then we define the prior distribution $P(H)$ for each of our two tasks, GENERIC and UPDATE summarization.

4.1 Extractive summarization algorithm

We first compute a surprise value for each word type in the summarization input. Word scores are aggregated to obtain a score for each sentence.

Step 1: Word score. Suppose that word type w_i appears c_i times in the summarization input I . We obtain the posterior distribution after seeing all instances of this word (\mathbf{w}_i) as $P(H|\mathbf{w}_i) = \text{Dir}(\alpha_1, \alpha_2, \dots, \alpha_i + c_i, \dots, \alpha_V)$. The score for w_i is the surprise computed as KL divergence between $P(H|\mathbf{w}_i)$ and the prior $P(H)$ (eqn. 6).

Step 2: Sentence score. The composition functions to obtain sentence scores from word scores can impact content selection performance (Nenkova et al., 2006). We experiment with sum and average value of the word scores.¹

Step 3: Sentence selection. The goal is to select a subset of sentences with high surprise values. We follow a greedy approach to optimize the summary surprise by choosing the most surprising sentence, the next most surprising and so on. At the same time, we aim to avoid redundancy, i.e. selecting sentences with similar content. After a sentence is selected for the summary, the surprise for words from this sentence are set to zero. We recompute the surprise for the remaining sentences using step 2 and the selection process continues until the summary length limit is reached.

The key differences between our Bayesian approach and a method such as topic words are: (i) The Bayesian approach keeps multiple hypotheses about the background rather than a single one. Surprise is computed based on the changes in probabilities of all of these hypotheses upon seeing the summarization input. (ii) The computation of topic words is local, it assumes a binomial distribution and the occurrence of a word is independent of others. In contrast, word surprise although computed for each word type separately, quantifies the surprise when incorporating the new counts of this word into the background multinomials.

4.2 Input and background

Here we describe the input sets and background corpus used for the two summarization tasks and

¹An alternative algorithm could directly compute the surprise of a sentence by incorporating the words from the sentence into the posterior. However, we found this specific method to not work well probably because the few and un-repeated content words from a sentence did not change the posterior much. In future, we plan to use latent topic models to assign a topic to a sentence so that the counts of all the sentence’s words can be aggregated into one dimension.

define the prior distribution for each. We use data from the DUC² and TAC³ summarization evaluation workshops conducted by NIST.

Generic summarization. We use multidocument inputs from DUC 2004. There were 50 inputs, each contains around 10 documents on a common topic. Each input is also provided with 4 manually written summaries created by NIST assessors. We use these manual summaries for evaluation.

The background corpus is a collection of 5000 randomly selected articles from the English Gigaword corpus. We use a list of 571 stop words from the SMART IR system (Buckley, 1985) and the remaining content word vocabulary has 59,497 word types. The count of each word in the background is calculated and used as the α parameters of the prior Dirichlet distribution $P(H)$ (eqn. 6).

Update summarization. This task uses data from TAC 2009. An input has two sets of documents, A and B, each containing 10 documents. Both A and B are on same topic but documents in B were published at a later time than A (background). There were 44 inputs and 4 manual update summaries are provided for each.

The prior parameters are the counts of words in A for that input (using the same stoplist). The vocabulary of these A sets is smaller, ranging from 400 to 3000 words for the different inputs.

In practice for both tasks, a new summarization input can have words unseen in the background. So *new* words in an input are added to the background corpus with a count of 1 and the counts of *existing* words in the background are incremented by 1 before computing the prior parameters. The summary length limit is 100 words in both tasks.

5 Systems for comparison

We compare against three types of systems, (i) those which similarly to surprise, use a background corpus to identify important sentences, (ii) a system that uses information from the input set only and no background, and (iii) systems that combine scores from the input and background.

KL_{back}: represents a simple baseline for surprise computation from a background corpus. A *single* unigram probability distribution B is created from the background using maximum likelihood. The summary is created by greedily adding sentences which maximize KL divergence

between B and the current summary. Suppose the set of sentences currently chosen in the summary is S . The next step chooses the sentence $s_l = \arg \max_{s_i} \text{KL}(\{S \cup s_i\} || B)$.

TS_{sum}, TS_{avg}: use topic words computed as described in Section 2 and utilizing the same background corpus for the generic and update tasks as the surprise-based methods. For the generic task, we use a critical value of 10 (0.001 significance level) for the χ^2 distribution during topic word computation. In the update task however, the background corpus A is smaller and for most inputs, no words exceeded this cutoff. We lower the significance level to the generally accepted value of 0.05 and take words scoring above this as topic words. The number of topic words is still small (ranging from 1 to 30) for different inputs.

The TS_{sum} system selects sentences with greater counts of topic words and TS_{avg} computes the number of topic words normalized by sentence length. A greedy selection procedure is used. To reduce redundancy, once a sentence is added, the topic words contained in it are removed from the topic word list before the next sentence selection.

KL_{inp}: represents the system that *does not use* background information. Rather the method creates a summary by optimizing for high similarity of the summary with the input word distribution.

Suppose the input unigram distribution is I and the current summary is S , the method chooses the sentence $s_l = \arg \min_{s_i} \text{KL}(\{S \cup s_i\} || I)$ at each iteration. Since $\{S \cup s_i\}$ is used to compute divergence, redundancy is implicitly controlled in this approach. Such a KL objective was used in competitive systems in the past (Daumé III and Marcu, 2006; Haghghi and Vanderwende, 2009).

Input + background: These systems combine (i) a score based on the background (KL_{back}, TS or SR) with (ii) the score based on the input only (KL_{inp}). For example, to combine TS_{sum} and KL_{inp}: for each sentence, we compute its scores based on the two methods. Then we normalize the two sets of scores for candidate sentences using z-scores and compute the best sentence as $\arg \max_{s_i} (\text{TS}_{\text{sum}}(s_i) - \text{KL}_{\text{inp}}(s_i))$. Redundancy control is done similarly to the TS only systems.

6 Content selection results

For evaluation, we compare each summary to the four manual summaries using ROUGE (Lin and Hovy, 2003; Lin, 2004). All summaries were truncated to 100 words, stemming was performed and

²<http://www-nlpir.nist.gov/projects/duc/index.html>

³<http://www.nist.gov/tac/>

	ROUGE-1	ROUGE-2
KL_{back}	0.2276 (TS, SR)	0.0250 (TS, SR)
TS_{sum}	0.3078	0.0616
TS_{avg}	0.2841 (TS_{sum} , SR_{sum})	0.0493 (TS_{sum})
SR_{sum}	0.3120	0.0580
SR_{avg}	0.3003	0.0549
KL_{inp}	0.3075 ($KL_{inp}+TS_{avg}$)	0.0684
$KL_{inp}+TS_{sum}$	0.3250	0.0725
$KL_{inp}+TS_{avg}$	0.3410	0.0795
$KL_{inp}+SR_{sum}$	0.3187 ($KL_{inp}+TS_{avg}$)	0.0660 ($KL_{inp}+TS_{avg}$)
$KL_{inp}+SR_{avg}$	0.3220 ($KL_{inp}+TS_{avg}$)	0.0696

Table 1: Evaluation results for generic summaries. Systems in parentheses are significantly better.

stop words were **not** removed, as is standard in TAC evaluations. We report the ROUGE-1 and ROUGE-2 recall scores (average over the inputs) for each system. We use the Wilcoxon signed-rank test to check for significant differences in mean scores. Table 1 shows the scores for generic summaries and 2 for the update task. For each system, the peer systems with significantly better scores (p -value < 0.05) are indicated within parentheses.

We refer to the surprise-based summaries as SR_{sum} and SR_{avg} depending on the type of composition function (Section 4.1).

First, consider GENERIC summarization and the systems which use the background corpus only (those above the horizontal line). The KL_{back} baseline performs significantly worse than topic words and surprise summaries. Numerically, SR_{sum} has the highest ROUGE-1 score and TS_{sum} tops according to ROUGE-2. As per the Wilcoxon test, TS_{sum} , SR_{sum} and SR_{avg} scores are statistically indistinguishable at 95% confidence level.

Systems below the horizontal line in Table 1 use an objective which combines both similarity with the input and difference from the background. The first line here shows that a system optimizing only for input similarity, KL_{inp} , by itself has higher scores (though not significant) than those using background information only. This result is not surprising for generic summarization where all the topical content is present in the input and the background is a non-focused random collection. At the same time, adding either TS or SR scores to KL_{inp} almost always leads to better results with $KL_{inp} + TS_{avg}$ giving the best score.

In UPDATE summarization, the surprise-based methods have an advantage over the topic word ones. SR_{avg} is significantly better than TS_{avg} for both ROUGE-1 and ROUGE-2 scores and better than TS_{sum} according to ROUGE-1. In fact, the surprise methods have numerically higher

	ROUGE-1	ROUGE-2
KL_{back}	0.2246 (TS, SR)	0.0213 (TS, SR)
TS_{sum}	0.3037 (SR_{avg})	0.0563
TS_{avg}	0.2909 (SR_{sum} , SR_{avg})	0.0477 (SR_{sum} , SR_{avg})
SR_{sum}	0.3201	0.0640
SR_{avg}	0.3226	0.0639
KL_{inp}	0.3098 ($KL_{inp}+SR_{avg}$)	0.0710
$KL_{inp}+TS_{sum}$	0.3010 ($KL_{inp}+SR_{sum, avg}$)	0.0635
$KL_{inp}+TS_{avg}$	0.3021 ($KL_{inp}+SR_{sum, avg}$)	0.0543 (KL_{inp} , $KL_{inp}+SR_{sum, avg}$)
$KL_{inp}+SR_{sum}$	0.3292	0.0721
$KL_{inp}+SR_{avg}$	0.3379	0.0767

Table 2: Evaluation results for update summaries. Systems in parentheses are significantly better.

ROUGE-1 scores compared to input similarity (KL_{inp}) in contrast to generic summarization. When combined with KL_{inp} , the surprise methods provide improved results, significantly better in terms of ROUGE-1 scores. The TS methods do not lead to any improvement, and $KL_{inp} + TS_{avg}$ is significantly worse than KL_{inp} only. The limitation of the TS approach arises from the paucity of topic words that exceed the significance cutoff applied on the log-likelihood ratio. But Bayesian surprise is robust on the small background corpus and does not need any tuning for cutoff values depending on the size of the background set.

Note that these models do not perform on par with summarization systems that use multiple indicators of content importance, involve supervised training and which perform sentence compression. Rather our goal in this work is to demonstrate a simple and intuitive unsupervised model.

7 Conclusion

We have introduced a Bayesian summarization method that strongly aligns with intuitions about how people use existing knowledge to identify important events or content in new observations.

Our method is especially valuable when a system must utilize a small background corpus. While the update task datasets we have used were carefully selected and grouped by NIST assessors into initial and background sets, for systems on the web, there is little control over the number of background documents on a particular topic. A system should be able to use smaller amounts of background information and as new data arrives, be able to incorporate the evidence. Our Bayesian approach is a natural fit in such a setting.

Acknowledgements

The author was supported by a Newton International Fellowship (NF120479) from the Royal Society and the British Academy.

References

- J. Allan, C. Wade, and A. Bolivar. 2003. Retrieval and novelty detection at the sentence level. In *Proceedings of SIGIR*, pages 314–321.
- P. Baldi and L. Itti. 2010. Of bits and wows: a bayesian theory of surprise with applications to attention. *Neural Networks*, 23(5):649–666.
- L. Bentivogli, P. Clark, I. Dagan, and D. Giampiccolo. 2010. The sixth PASCAL recognizing textual entailment challenge. *Proceedings of TAC*.
- C. Buckley. 1985. Implementation of the SMART information retrieval system. Technical report, Cornell University.
- J. Conroy, J. Schlesinger, and D. O’Leary. 2006. Topic-focused multi-document summarization using an approximate oracle score. In *Proceedings of COLING-ACL*, pages 152–159.
- H. Daumé III and D. Marcu. 2006. Bayesian query-focused summarization. In *Proceedings of ACL*, pages 305–312.
- J. Delort and E. Alfonseca. 2012. DualSum: A topic-model based approach for update summarization. In *Proceedings of EACL*, pages 214–223.
- A. Haghighi and L. Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of NAACL-HLT*, pages 362–370.
- L. Itti and P. F. Baldi. 2006. Bayesian surprise attracts human attention. In *Proceedings of NIPS*, pages 547–554.
- C. Lin and E. Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *Proceedings of COLING*, pages 495–501.
- C. Lin and E. Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of HLT-NAACL*, pages 1085–1090.
- C. Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Proceedings of Text Summarization Branches Out Workshop, ACL*, pages 74–81.
- A. Nenkova, L. Vanderwende, and K. McKeown. 2006. A compositional context sensitive multi-document summarizer: exploring the factors that influence summarization. In *Proceedings of SIGIR*, pages 573–580.
- W. D Penny. 2001. Kullback-liebler divergences of normal, gamma, dirichlet and wishart densities. *Wellcome Department of Cognitive Neurology*.
- B. Schiffman. 2005. *Learning to Identify New Information*. Ph.D. thesis, Columbia University.
- I. Soboroff and D. Harman. 2005. Novelty detection: the trec experience. In *Proceedings of HLT-EMNLP*, pages 105–112.

Predicting Power Relations between Participants in Written Dialog from a Single Thread

Vinodkumar Prabhakaran

Dept. of Computer Science
Columbia University, New York, NY
vinod@cs.columbia.edu

Owen Rambow

Cntr. for Comp. Learning Systems
Columbia University, New York, NY
rambow@ccls.columbia.edu

Abstract

We introduce the problem of predicting who has power over whom in pairs of people based on a single written dialog. We propose a new set of structural features. We build a supervised learning system to predict the direction of power; our new features significantly improve the results over using previously proposed features.

1 Introduction

Computationally analyzing the social context in which language is used has gathered great interest within the NLP community recently. One of the areas that has generated substantial research is the study of how social power relations between people affect and/or are revealed in their interactions with one another. Researchers have proposed systems to detect social power relations between participants of organizational email threads (Bramsen et al., 2011; Gilbert, 2012; Prabhakaran and Rambow, 2013), online forums (Danescu-Niculescu-Mizil et al., 2012; Biran et al., 2012; Danescu-Niculescu-Mizil et al., 2013), chats (Strzalkowski et al., 2012), and off-line interactions such as presidential debates (Prabhakaran et al., 2013; Nguyen et al., 2013). Automatically identifying power and influence from interactions can have many practical applications ranging from law enforcement and intelligence to online marketing.

A significant number of these studies are performed in the domain of organizational email where there is a well defined notion of power (organizational hierarchy). Bramsen et al. (2011) and Gilbert (2012) predict hierarchical power relations between people in the Enron email corpus using lexical features extracted from *all* the messages exchanged between them. However, their approaches primarily apply to situations where large collections of messages exchanged between pairs

of people are available. In (Prabhakaran and Rambow, 2013), we introduced the problem of detecting whether a participant of an email thread has power over someone else in the thread and established the importance of dialog structure in that task. However, in that work we did not detect over whom that person has power.

In this paper, we introduce a new problem formulation. We predict the hierarchical power relation between pairs of participants in an email interaction thread based *solely* on features extracted from that thread. As a second major contribution, we introduce a new set of features to capture aspects of participant behavior such as responsiveness, and we show that these features are significantly correlated with the direction of power. We present a fully automatic system for this task obtaining an accuracy of 73.0%, an improvement of 6.9% over 68.3% by a system using only lexical features. This best-performing system uses our new feature set.

2 Motivation

Early NLP-based approaches such as Bramsen et al. (2011) and Gilbert (2012) built systems to predict hierarchical power relations between people in the Enron email corpus using lexical features from all the messages exchanged between them. One limitation of this approach is that it relies solely on lexical cues and hence works best when large collections of messages exchanged between the pairs of people are available. For example, Bramsen et al. (2011) excluded sender-recipient pairs who exchanged fewer than 500 words from their evaluation set, since they found smaller text samples are harder to classify. By taking the message out of the context of the interaction in which it was exchanged, they fail to utilize cues from the structure of interactions, which complements the lexical cues in detecting power relations, as we showed in (Prabhakaran and Rambow, 2013).

We modeled the problem of detecting power relationships differently in (Prabhakaran and Rambow, 2013): we predicted whether a participant in an email thread has a certain type of power or not. However, in that work we did not predict over whom he/she has that power. This may result in noisy features; consider a thread in which participant \mathcal{X} has power over participant \mathcal{Y} , who has power over participant \mathcal{Z} . By aggregating features over all messages sent by \mathcal{Y} , features salient to a subordinate-superior interaction are incorrectly conflated with those salient to superior-subordinate interaction. Another limitation of (Prabhakaran and Rambow, 2013) is that we used manual annotations for many of our features such as dialog acts and overt displays of power. Relying on manual annotations for features limited our analysis to a small subset of the Enron corpus, which has only 18 instances of hierarchical power. Consequently, our findings with respect to hierarchical power were weak in terms of both correlations of features and system performance.

In this paper, we introduce the problem of predicting who has power over whom in pairs of interacting participants based on a single thread of interactions. From (Bramsen et al., 2011) we retain the idea that we want to predict the power relation between pairs of people. But in contrast to their formulation, we retain the goal from (Prabhakaran and Rambow, 2013) that we want to study communication in the context of an interaction, and that we want to be able to make predictions using only the emails exchanged in a single thread. Like (Prabhakaran and Rambow, 2013), we use features to capture the dialog structure, but we use automatic taggers to generate them and assume no manual annotation at all at training or test time. This allows us to use the entire Enron email corpus for this study.

3 Data

In this work, we use the version of Enron email corpus by Yeh and Harnly (2006) which captures the thread structure of email exchanges. The corpus contains 36,615 email threads. We excluded a small subset of 419 threads that was used for previous manual annotation efforts, part of which was also used to train the DA and ODP taggers (Section 5) that generate features for our system. The average number of email messages per thread was around 3. We divided the remaining threads into

train (50%), *dev* (25%) and *test* (25%) sets by random sampling. We then applied various basic NLP preprocessing steps such as tokenization, POS tagging and lemmatization to the body of email messages. We use the Enron gold organizational hierarchy released by Agarwal et al. (2012) to model hierarchical power. Their corpus was manually built using information from Enron organizational charts. It includes relations of 1,518 employees and captures dominance relations between 13,724 pairs of them. This is the largest such data set available to the best of our knowledge.

4 Problem Formulation

Let t denote an email thread and M_t denote the set of all messages in t . Also, let P_t be the set of all participants in t , i.e., the union of senders and recipients (*To* and *CC*) of all messages in M_t . We are interested in detecting power relations between pairs of participants who interact within a given email thread. Not every pair of participants $(p_1, p_2) \in P_t \times P_t$ interact with one another within t . Let $IM_t(p_1, p_2)$ denote the set of *Interaction Messages* — non-empty messages in t in which either p_1 is the sender and p_2 is one of the recipients or vice versa. We call the set of (p_1, p_2) such that $|IM_t(p_1, p_2)| > 0$ the *interacting participant pairs* of t (IPP_t).

We focus on the manifestations of power in interactions between people across different levels of hierarchy. For every $(p_1, p_2) \in IPP_t$, we query the set of dominance relations in the gold hierarchy to determine their hierarchical power relation ($HP(p_1, p_2)$). We exclude pairs that do not exist in the gold hierarchy from our analysis and denote the remaining set of *related interacting participant pairs* as $RIPP_t$. We assign $HP(p_1, p_2)$ to be *superior* if p_1 dominates p_2 , and *subordinate* if p_2 dominates p_1 . Table 1 shows the total number of pairs in IPP_t and $RIPP_t$ from all the threads in our corpus and across *train*, *dev* and *test* sets.

Description	Total	Train	Dev	Test
# of threads	36,196	18,079	8,973	9,144
$\sum_t IPP_t $	355,797	174,892	91,898	89,007
$\sum_t RIPP_t $	15,048	7,510	3,578	3,960

Table 1: Data Statistics
Row 1 presents the total number of threads in different subsets of the corpus. Row 2 and 3 present the number of interacting participant pairs (IPP) and related interacting participant pairs ($RIPP$) in those subsets.

Given a thread t and a pair of participants $(p_1, p_2) \in RIPP_t$, we want to automatically detect $HP(p_1, p_2)$. This problem formulation is similar to the ones in (Bramsen et al., 2011) and (Gilbert, 2012). However, the difference is that for us an instance is a pair of participants in a single thread of interaction (which may or may not include other people), whereas for them an instance constitutes all messages exchanged between a pair of people in the entire corpus. Our formulation also differs from (Prabhakaran and Rambow, 2013) in that we detect power relations between pairs of participants, instead of just whether a participant had power over anyone in the thread.

5 Structural Analysis

In this section we analyze various features that capture the structure of interaction between the pairs of participants in a thread. Each feature f is extracted with respect to a person p over a reference set of messages M (denoted f_M^p). For a pair (p_1, p_2) , we extract 4 versions of each feature f : $f_{IM_t(p_1, p_2)}^{p_1}$, $f_{IM_t(p_1, p_2)}^{p_2}$, $f_{M_t}^{p_1}$ and $f_{M_t}^{p_2}$. The first two capture behavior of the pair among themselves, while the third and fourth capture their overall behavior in the entire thread. We group our features into three categories — THR^{New} , THR^{PR} and DIA^{PR} . THR^{New} is a set of new features we propose, while THR^{PR} and DIA^{PR} incorporate features we proposed in (Prabhakaran and Rambow, 2013). THR^{New} and THR^{PR} capture the structure of message exchanges without looking at the content of the emails (e.g., how many emails did a person send), while DIA^{PR} captures the pragmatics of the dialog and requires an analysis of the content of the emails (e.g., did they issue any requests).

THR^{New} : This is a new set of features we introduce in this paper. It includes the average number of recipients (AvgRecipients) and *To* recipients (AvgToRecipients) in emails sent by p , the percentage of emails p received in which he/she was in the *To* list (InToList%), boolean features denoting whether p added or removed people when responding to a message (AddPerson and RemovePerson), average number of replies received per message sent by p (ReplyRate) and average number of replies received from the other person of the pair to messages where he/she was a *To* recipient (ReplyRateWithinPair). ReplyRateWithinPair applies only to $IM_t(p_1, p_2)$.

THR^{PR} : This feature set includes two meta-

data based feature sets — positional and verbosity. Positional features include a boolean feature to denote whether p sent the first message (Initiate), and relative positions of p 's first and last messages (FirstMsgPos and LastMsgPos) in M . Verbosity features include p 's message count (MsgCount), message ratio (MsgRatio), token count (TokenCount), token ratio (TokenRato) and tokens per message (TokenPerMsg), all calculated over M .

DIA^{PR} : In (Prabhakaran and Rambow, 2013), we used dialog features derived from manual annotations — dialog acts (DA) and overt displays of power (ODP) — to model the structure of interactions within the message content. In this work, we obtain DA and ODP tags on the entire corpus using automatic taggers trained on those manual annotations. The DA tagger (Omuya et al., 2013) obtained an accuracy of 92%. The ODP tagger (Prabhakaran et al., 2012) obtained an accuracy of 96% and F-measure of 54%. The DA tagger labels each sentence to be one of the 4 dialog acts: Request Action, Request Information, Inform, and Conventional. The ODP Tagger identifies sentences (mostly requests) that express additional constraints on its response, beyond those introduced by the dialog act. We use 5 features: ReqAction%, ReqInform%, Inform%, Conventional%, and ODP% to capture the percentage of sentences in messages sent by p that has each of these labels. We also use a feature to capture the number of p 's messages with a request that did not get a reply, i.e., dangling requests (DanglingReq%), over all messages sent by p .

We perform an unpaired two-sample two-tailed Student's t-Test comparing mean values of each feature for *subordinates* vs. *superiors*. For our analysis, a data point is a related interacting pair, and not a message. Hence, a message with multiple recipients who have a *superior/subordinate* relation with the sender will contribute to features for multiple data points. We limit our analysis to the related interacting pairs from only our *train* set. Table 2 presents mean values of features for *subordinates* and *superiors* at the interaction level. Thread level versions of these features also obtained similar results overall in terms of direction of difference and significance. We denote three significance levels — * ($p < .05$), ** ($p < .01$), and *** ($p < .001$). To control false discovery rates in multiple testing, we adjusted the p-values (Benjamini and Hochberg, 1995). We summarize

Feature Name	Mean($f_{IM_t}^{sub}$)	Mean($f_{IM_t}^{sup}$)
<i>THR^{New}</i>		
AvgRecipients***	21.14	43.10
AvgToRecipients***	18.19	38.94
InToList%	0.82	0.80
ReplyRate***	0.86	1.23
ReplyRateWithinPair***	0.16	0.10
AddPerson	0.48	0.47
RemovePerson***	0.41	0.37
<i>THR^{PR}</i>		
Initiate***	0.45	0.56
FirstMsgPos	0.04	0.03
LastMsgPos***	0.15	0.11
MsgCount***	0.64	0.70
MsgRatio***	0.44	0.56
TokenCount	91.22	83.26
TokenRatio***	0.45	0.55
TokenPerMsg*	140.60	120.87
<i>DIA^{PR}</i>		
Conventional%***	0.15	0.17
Inform%***	0.78	0.72
ReqAction%***	0.02	0.04
ReqInform%***	0.05	0.06
DanglingReq%***	0.12	0.15
ODP%***	0.03	0.06

Table 2: Student’s t-Test Results of $f_{IM_t}^P$.
 THR^{New} : new meta-data features; THR^{PR} , DIA^{PR} : meta-data
and dialog-act features from previous studies;
* ($p < .05$); ** ($p < .01$); *** ($p < .001$)

the main findings on the significant features below.

1. Superiors send messages addressed to more people (AvgRecipients and AvgToRecipients). Consequently, they get more replies to their messages (ReplyRate). However, considering messages where the other person of the pair is addressed in the *To* list (ReplyRate-WithinPair), subordinates get more replies.
2. Superiors issue more requests (ReqAction% and ReqInform%) and overt displays of power (ODP%). Subordinates issue more informs (Inform%) and, surprisingly, have fewer unanswered requests (DanglingReq%).
3. Superiors initiate the interactions more often than subordinates (Initiate). They also leave interactions earlier (LastMsgPos).
4. Superiors send shorter messages (TokenPerMsg). They also send more messages (MsgCount & MsgRatio) and even contribute a higher ratio of tokens in the thread (TokenRatio) despite sending shorter messages.

Finding 1 goes in line with findings from studies analyzing social networks that superiors have higher connectivity in the networks that they are part of (Rowe et al., 2007). Intuitively, those who have higher connectivity also send emails to larger number of people, and hence our result. Since superiors address more people in their emails, they also have a higher chance of getting replies. Finding 2 also aligns with the general intuition about how superiors and subordinates behave within interactions (e.g., superiors exhibit more overt displays of power than subordinates).

Findings 3 & 4 are interesting since they reveal special characteristics of threads involving hierarchically related participants. In (Prabhakaran and Rambow, 2013), we had found that persons with hierarchical power rarely initiated threads and contributed less within the threads. But that problem formulation was different — we were identifying whether a person in a given thread had hierarchical power over someone else or not. The data points in that formulation included participants from threads that did not have any hierarchically related people, whereas our current formulation do not. These findings suggest that if a person starts an email thread, he’s likely not to be the one who has power, but if a thread includes a pair of people who are hierarchically related, then it is likely to be initiated by the superior and he/she tends to contribute more in such threads.

6 Predicting Direction of Power

We build an SVM-based supervised learning system that can predict $HP(p_1, p_2)$ to be either *superior* or *subordinate* based on the interaction within a thread t for any pair of participants $(p_1, p_2) \in RIPP_t$. We deterministically fix the order of participants in (p_1, p_2) such that p_1 is the sender of the first message in $IM_t(p_1, p_2)$. We use the ClearTK (Ogren et al., 2008) wrapper for SVMLight (Joachims, 1999) in our experiments. We use the related interacting participant pairs in threads from the *train* set to train our models and optimize our performance on those from the *dev* set. We report results obtained on *dev* and *test* sets.

In our formulation, values of many features are undefined for some instances (e.g., Inform% is undefined when MsgCount = 0). Handling of undefined values for features in SVM is not straightforward. Most SVM implementations assume the value of 0 by default in such cases, conflating them

Description	Accuracy
Baseline (Always Superior)	52.54
Baseline (Word Unigrams + Bigrams)	68.56
THR ^{New}	55.90
THR ^{PR}	54.30
DIA ^{PR}	54.05
THR ^{PR} + THR ^{New}	61.49
DIA ^{PR} + THR ^{PR} + THR ^{New}	62.47
LEX	70.74
LEX + DIA ^{PR} + THR ^{PR}	67.44
LEX + DIA ^{PR} + THR ^{PR} + THR ^{New}	68.56
BEST (= LEX + THR ^{New})	73.03
BEST (Using p_1 features only)	72.08
BEST (Using IM_t features only)	72.11
BEST (Using M_t only)	71.27
BEST (No Indicator Variables)	72.44

Table 3: Accuracies on feature subsets (*dev* set). THR^{New}: new meta-data features; THR^{PR}, DIA^{PR}: meta-data and dialog-act features from previous studies; LEX: ngrams; BEST: best subset; IM_t stands for $IM_t(p_1, p_2)$

with cases where Inform% is truly 0. In order to mitigate this issue, we use an indicator feature for each structural feature to denote whether or not it is valid. Since we use a quadratic kernel, we expect the SVM to pick up the interaction between each feature and its indicator feature.

Lexical features have already been shown to be valuable in predicting power relations (Bramsen et al., 2011; Gilbert, 2012). We use another feature set LEX to capture word ngrams, POS (part of speech) ngrams and mixed ngrams. A mixed ngram (Prabhakaran et al., 2012) is a special case of word ngram where words belonging to open classes are replaced with their POS tags. We found the best setting to be using both unigrams and bigrams for all three types of ngrams, by tuning in our *dev* set. We then performed experiments using all subsets of {LEX, THR^{New}, THR^{PR}, DIA^{PR}}.

Table 3 presents the results obtained using various feature subsets. We use a majority class baseline assigning $HP(p_1, p_2)$ to be always *superior*, which obtains 52.5% accuracy. We also use a stronger baseline using word unigrams and bigrams as features, which obtained an accuracy of 68.6%. The performance of the system using each structural feature class on its own is very low. Combining all three of them improves the accuracy to 62.5%. The highest performance obtained without using any message content is for THR^{PR} and THR^{New} (61.5%). LEX features by

itself obtain a very high accuracy of 70.7%, confirming the importance of lexical patterns in this task. Perplexingly, adding all structural features to LEX reduces the accuracy by around 2.2 percentage points. The best performing system (BEST) uses LEX and THR^{New} features and obtains an accuracy of 73.0%, a statistically significant improvement over the LEX-only system (McNemar).

We also performed an ablation study to understand the importance of different slices of our feature sets. If we remove all feature versions with respect to the second person, the accuracy drops to 72.1%. This suggests that features about the other person’s behavior also help the prediction task. If we remove either the thread level versions of features or interaction level versions of features, the accuracy again drops, suggesting that both the pair’s behavior among themselves, and their overall behavior in the thread add value to the prediction task. Removing the indicator feature denoting the structural features’ validity also reduces the performance of the system.

We now discuss evaluation on our blind test set. The majority baseline (Always Superior) for accuracy is 55.0%. The word unigrams and bigrams baseline obtains an accuracy of 68.3%. The LEX system (using other forms of ngrams as well) obtains a slightly lower accuracy of 68.1%. Our BEST system using LEX and THR^{New} features obtains an accuracy of 73.0% (coincidentally the same as on the *dev* set), an improvement of 6.9% over the system using only lexical features.

7 Conclusion

We introduced the problem of predicting who has power over whom based on a single thread of written interactions. We introduced a new set of features which describe the structure of the dialog. Using this feature set, we obtain an accuracy of 73.0% on a blind test. In future work, we will tackle the problem of three-way classification of pairs of participants, which will cover cases in which they are not in a power relation at all.

Acknowledgments

This paper is based upon work supported by the DARPA DEFT Program. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government. We also thank several anonymous reviewers for their feedback.

References

- Apoorv Agarwal, Adinoyi Omuya, Aaron Harnly, and Owen Rambow. 2012. A Comprehensive Gold Standard for the Enron Organizational Hierarchy. In *Proceedings of the 50th Annual Meeting of the ACL (Short Papers)*, pages 161–165, Jeju Island, Korea, July. Association for Computational Linguistics.
- Yoav Benjamini and Yoel Hochberg. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 289–300.
- Or Biran, Sara Rosenthal, Jacob Andreas, Kathleen McKeown, and Owen Rambow. 2012. Detecting influencers in written online conversations. In *Proceedings of the Second Workshop on Language in Social Media*, pages 37–45, Montréal, Canada, June. Association for Computational Linguistics.
- Philip Bramsen, Martha Escobar-Molano, Ami Patel, and Rafael Alonso. 2011. Extracting social power relationships from natural language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 773–782, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Cristian Danescu-Niculescu-Mizil, Lillian Lee, Bo Pang, and Jon Kleinberg. 2012. Echoes of power: language effects and power differences in social interaction. In *Proceedings of the 21st international conference on World Wide Web, WWW '12*, New York, NY, USA. ACM.
- Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A computational approach to politeness with application to social factors. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 250–259, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Eric Gilbert. 2012. Phrases that signal workplace hierarchy. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work, CSCW '12*, pages 1037–1046, New York, NY, USA. ACM.
- Thorsten Joachims. 1999. Making Large-Scale SVM Learning Practical. In Bernhard Schölkopf, Christopher J.C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, Cambridge, MA, USA. MIT Press.
- Viet-An Nguyen, Jordan Boyd-Graber, Philip Resnik, Deborah A. Cai, Jennifer E. Midberry, and Yuanxin Wang. 2013. Modeling topic control to detect influence in conversations using nonparametric topic models. *Machine Learning*, pages 1–41.
- Philip V. Ogren, Philipp G. Wetzler, and Steven Bethard. 2008. ClearTK: A UIMA toolkit for statistical natural language processing. In *Towards Enhanced Interoperability for Large HLT Systems: UIMA for NLP workshop at Language Resources and Evaluation Conference (LREC)*.
- Adinoyi Omuya, Vinodkumar Prabhakaran, and Owen Rambow. 2013. Improving the quality of minority class identification in dialog act tagging. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 802–807, Atlanta, Georgia, June. Association for Computational Linguistics.
- Vinodkumar Prabhakaran and Owen Rambow. 2013. Written dialog and social power: Manifestations of different types of power in dialog behavior. In *Proceedings of the IJCNLP*, pages 216–224, Nagoya, Japan, October. Asian Federation of Natural Language Processing.
- Vinodkumar Prabhakaran, Owen Rambow, and Mona Diab. 2012. Predicting Overt Display of Power in Written Dialogs. In *Human Language Technologies: The 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Montreal, Canada, June. Association for Computational Linguistics.
- Vinodkumar Prabhakaran, Ajita John, and Dorée D. Seligmann. 2013. Who had the upper hand? ranking participants of interactions based on their relative power. In *Proceedings of the IJCNLP*, pages 365–373, Nagoya, Japan, October. Asian Federation of Natural Language Processing.
- Ryan Rowe, German Creamer, Shlomo Hershkop, and Salvatore J. Stolfo. 2007. Automated social hierarchy detection through email network analysis. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web Mining and Social Network Anal.* ACM.
- Tomek Strzalkowski, Samira Shaikh, Ting Liu, George Aaron Broadwell, Jenny Stromer-Galley, Sarah Taylor, Umit Boz, Veena Ravishankar, and Xiaoai Ren. 2012. Modeling leadership and influence in multi-party online discourse. In *Proceedings of COLING*, pages 2535–2552, Mumbai, India, December. The COLING 2012 Organizing Committee.
- Jen-Yuan Yeh and Aaron Harnly. 2006. Email thread reassembly using similarity matching. In *CEAS 2006 - The Third Conference on Email and Anti-Spam, July 27-28, 2006, Mountain View, California, USA*, Mountain View, California, USA, July.

Tri-Training for Authorship Attribution with Limited Training Data

Tieyun Qian

State Key Laboratory
of Software Eng.,
Wuhan University
430072, Hubei, China
qty@whu.edu.cn

Bing Liu

Dept. of Computer Sci-
ence, Univ. of Illinois at
Chicago
IL, USA, 60607
liub@cs.uic.edu

Li Chen

State Key Laboratory of
Software Eng.,
Wuhan University
430072, Hubei, China
ccnuchenli@163.com

Zhiyong Peng

Computer School,
Wuhan University
430072, Hubei, China
peng@whu.edu.cn

Abstract

Authorship attribution (AA) aims to identify the authors of a set of documents. Traditional studies in this area often assume that there are a large set of labeled documents available for training. However, in the real life, it is often difficult or expensive to collect a large set of labeled data. For example, in the online review domain, most reviewers (authors) only write a few reviews, which are not enough to serve as the training data for accurate classification. In this paper, we present a novel three-view tri-training method to iteratively identify authors of unlabeled data to augment the training set. The key idea is to first represent each document in three distinct views, and then perform tri-training to exploit the large amount of unlabeled documents. Starting from 10 training documents per author, we systematically evaluate the effectiveness of the proposed tri-training method for AA. Experimental results show that the proposed approach outperforms the state-of-the-art semi-supervised method CNG+SVM and other baselines.

1 Introduction

Existing approaches to authorship attribution (AA) are mainly based on supervised classification (Stamatatos, 2009, Kim et al., 2011, Serousi et al., 2012). Although this is an effective approach, it has a major weakness, i.e., for each author a large number of his/her articles are needed as the training data. This is possible if the author has written a large number of articles, but will be difficult if he/she has not. For example, in the online review domain, most authors (reviewers) only write a few reviews (documents). It was shown that on average each reviewer only has 2.72 reviews in amazon.com, and only 8% of the reviewers have at least 5 reviews (Jindal and Liu, 2008). The small number of labeled documents makes it extremely challenging for supervised

learning to train an accurate classifier.

In this paper, we consider AA with only a few labeled examples. By exploiting the redundancy in human languages, we tackle the problem using a new three-view tri-training algorithm (TTA). Specifically, we first represent each document in three distinct views, and then tri-train three classifiers in these views. The predictions of two classifiers on unlabeled examples are used to augment the training set for the third classifier. This process repeats until a termination condition is met. The enlarged labeled sets are finally used to train classifiers to classify the test data.

To our knowledge, no existing work has addressed AA in a tri-training framework. The AA problem with limited training data was attempted in (Stamatatos, 2007; Luyckx and Daelemans, 2008). However, neither of them used a semi-supervised approach to augment the training set with additional documents. Kourtis and Stamatatos (2011) introduced a variant of the self-training method in (Nigam and Ghani, 2000). Note that the original self-training uses one classifier on one view. However, the self-training method in (Kourtis and Stamatatos, 2011) uses two classifiers (CNG and SVM) on one view. Both the self-training and tri-training are semi-supervised learning methods. However, the proposed approach is not a simple extension of the self-training method CNG+SVM of (Kourtis and Stamatatos, 2011). There are key differences.

First, in their experimental setting, about 115 and 129 documents per author on average are used for two experimental corpora. This number of labeled documents is still very large. We consider a much more realistic problem, where the size of the training set is very small. Only 10 samples per author are used in training.

Second, CNG+SVM uses two learning methods on a single character n-gram view. In contrast, besides the character n-gram view, we also make use of the lexical and syntactic views. That is,

three distinct views are used for building classifiers. The redundant information in human language is combined in the tri-training procedure.

Third, in each round of self-training in CNG+SVM, each classifier is refined by the same newly labeled examples. However, in the proposed tri-training method (TTA), the examples labeled by the classifiers of every two views are added to the third view. By doing so, each classifier can borrow information from the other two views. And the predictions made by two classifiers are more reliable than those by one classifier.

The main contribution of this paper is thus the proposed three-view tri-training scheme which has a much better generalization ability by exploiting three different views of the same document. Experimental results on the IMDb review dataset show that the proposed method dramatically improves the CNG+SVM method. It also outperforms the co-training method (Blum and Mitchell, 1998) based on our proposed views.

2 Related Work

Existing AA methods either focused on finding suitable features or on developing effective techniques. Example features include function words (Argamon et al., 2007), richness features (Gamon 2004), punctuation frequencies (Graham et al., 2005), character (Grieve, 2007), word (Burrows, 1992) and POS n-grams (Gamon, 2004; Hirst and Feiguina, 2007), rewrite rules (Halteren et al., 1996), and similarities (Qian and Liu, 2013). On developing effective learning techniques, supervised classification has been the dominant approach, e.g., neural networks (Graham et al., 2005; Zheng et al., 2006), decision tree (Uzuner and Katz, 2005; Zhao and Zobel, 2005), logistic regression (Madigan et al., 2005), SVM (Diederich et al., 2000; Gamon 2004; Li et al., 2006; Kim et al., 2011), etc.

The main problem in the traditional research is the unrealistic size of the training set. A size of about 10,000 words per author is regarded as a reasonable training set size (Argamon et al., 2007, Burrows, 2003). When no long documents are available, tens or hundreds of short texts are used (Halteren, 2007; Hirst and Feiguina, 2007; Schwartz et al., 2013).

Apart from the existing works dealing with limited data discussed in the introduction, our preliminary study in (Qian et al., 2014) used one learning method on two views, but it is inferior to the proposed method in this paper.

Input: A small set of labeled documents $L = \{l_1, \dots, l_r\}$, a large set of unlabeled documents $U = \{u_1, \dots, u_s\}$, and a set of test documents $T = \{t_1, \dots, t_i\}$,

Parameters: the number of iterations k , the size of selected unlabeled documents u

Output: t_k 's class assignment

```

1 Extract views  $L_c, L_l, L_s, U_c, U_l, U_s, T_c, T_l, T_s$  from  $L, U, T$ 
2 Loop for  $k$  iterations:
3   Randomly select  $u$  unlabeled documents  $U'$  from  $U$ ;
4   Learn the first view classifier  $C_1$  from  $L_1$  ( $L_1=L_c, L_l$ , or  $L_s$ );
5   Use  $C_1$  to label docs in  $U'$  based on  $U_1$  ( $U_1=U_c, U_l$ , or  $U_s$ )
6   Learn the second view classifier  $C_2$  from  $L_2$  ( $L_2 \neq L_1$ )
7   Use  $C_2$  to label documents in  $U'$  based on  $U_2$  ( $U_2 \neq U_1$ );
8   Learn the third view classifier  $C_3$  from  $L_3$  ( $L_2 \neq L_1, L_2$ )
9   Use  $C_3$  to label documents in  $U'$  based on  $U_3$  ( $U_2 \neq U_1, U_2$ );
10   $U_{p1} = \{u \mid u \in U', u.\text{label by } C_2 = u.\text{label by } C_3\}$ ;
11   $U_{p2} = \{u \mid u \in U', u.\text{label by } C_1 = u.\text{label by } C_3\}$ ;
12   $U_{p3} = \{u \mid u \in U', u.\text{label by } C_1 = u.\text{label by } C_2\}$ ;
13   $U = U - U', L_i = L_i \cup U_{pi}$  ( $i=1..3$ );
14 Learn three classifiers  $C_1, C_2, C_3$  from  $L_1, L_2, L_3$ ;
15 Use  $C_i$  to label  $t_k$  in  $T_i$  ( $i=1..3$ );
16 Aggregate results from three views

```

Figure 1: The tri-training algorithm (TTA)

3 Proposed Tri-Training Algorithm

3.1 Overall Framework

We represent each document in three feature views: the character view, the lexical view and the syntactic view. Each view consists of a set of features in the respective type. A classifier can be learned from any of these views. We propose a three-view training algorithm to deal with the problem of limited training data. Logistic regression (LR) is used as the learner. The overall framework is shown in Figure 1.

Given the labeled, unlabeled, and test sets L, U , and T , step 1 extracts the character, lexical, and syntactic views from L, U , and T , respectively. Steps 2-13 iteratively tri-train three classifiers by adding the data which are assigned the same label by two classifiers into the training set of the third classifier. The algorithm first randomly selects u unlabeled documents from U to create a pool U' of examples. Note that we can directly select from the large unlabeled set U . However, it is shown in (Blum and Mitchell 2008) that a smaller pool can force the classifiers to select instances that are more representative of the underlying distribution that generates U . Hence we set the parameter u to a size of about 1% of the whole unlabeled set, which allows us to observe the effects of different number of iterations. It then iterates over the following steps. First, use character, lexical and syntactic views on the current labeled set to train three classifiers C_1, C_2 , and C_3 . See Steps 4-9. Second,

allow two of these three classifiers to classify the unlabeled set U' and choose p documents with agreed labels. See Steps 10-12. The selected documents are then added to the third labeled set for the label assigned (a label is an author here), and the u documents are removed from the unlabeled pool U' (line 13). We call this way of augmenting the training sets *InterAdding*. The one used in (Kourtis and Stamatatos, 2011) is called *SelfAdding* as it uses only a single view and adds to the same training set. Steps 14-15 assign the test document to a category (author) using the classifier learned from the three views in the augmented labeled data, respectively. Step 16 aggregates the results from three classifiers.

3.2 Character View

The features in the character view are the character n -grams of a document. Character n -grams are simple and easily available for any natural language. For a fair comparison with the previous work in (Kourtis and Stamatatos, 2011), we extract frequencies of 3-grams at the character-level. The vocabulary size for character 3-grams in our experiment is 28584.

3.3 Lexical View

The lexical view consists of word unigrams of a document. We represent each article by a vector of word frequencies. The vocabulary size for unigrams in our experiment is 195274.

3.4 Syntactic View

The syntactic view consists of the syntactic features of a document. We use four content-independent structures including n -grams of POS tags ($n = 1..3$) and rewrite rules (Kim et al., 2011). The vocabulary sizes for POS 1-grams, POS 2-grams, POS 3-grams, and rewrite rules in our experiment are 63, 1917, 21950, and 19240, respectively. These four types of syntactic structures are merged into a single vector. Hence the syntactic view of a document is represented as a vector of 43140 components.

3.5 Aggregating Results from Three Views

In testing, once we obtain the prediction values from three classifiers for a test document t_k , an additional algorithm is used to decide the final author attribution. One simple method is voting. However, this method is weaker than the three methods below. It is also hard to compare with the self-training method CNG+SVM in (Kourtis and Stamatatos, 2011) as it only has two classifi-

ers. Hence we present three other strategies to further aggregate the results from the three views. These methods require the classifier to produce a numeric score to reflect the positive or negative certainty. Many classification algorithms give such scores, e.g., SVM and logistic regression. The three methods are as follows:

- 1) *ScoreSum*: The learned model first classifies all test cases in T . Then for each test case t_k , this method sums up all scores of positive classifications from the three views. It then assigns t_k to the author with the highest score.
- 2) *ScoreSqSum*: This method works similarly to *ScoreSum* above except that it sums up the squared scores of positive classifications.
- 3) *ScoreMax*: This method works similarly to the *ScoreSum* method as well except that it finds the maximum classification score for each test document.

4 Experimental Evaluation

We now evaluate the proposed method. We use logistic regression (LR) with L2 regularization (Fan et al., 2008) and the *SVM^{multiclass}* (SVM) system (Joachims, 2007) with its default settings as the classifiers.

4.1 Experiment Setup

We conduct experiments on the IMDb dataset (Seroussi et al., 2010). This data set contains the IMDb reviews in May 2009. It has 62,000 reviews by 62 users (1,000 reviews per user). For each author/reviewer, we further split his/her documents into the labeled, unlabeled, and test sets. 1% of one author's documents, i.e., 10 documents per author, are used as the labeled data for training, 79% are used as unlabeled data, and the rest 20% are used for testing. We extract and compute the character and lexical features directly from the raw data, and use the Stanford PCFG parser (Klein and Manning, 2003) to generate the grammar structures of sentences in each review for extracting syntactic features. We normalize each feature's value to the $[0, 1]$ interval by dividing the maximum value of this feature in the training set. We use the micro-averaged classification accuracy as the evaluation metric.

4.2 Baseline methods

We use six self-training baselines and three co-training baselines. Self-training in (Kourtis and Stamatatos, 2011) uses two different classifiers on one view, and co-training uses one classifier on two views. All baselines except CNG+SVM

on the character view are our extensions.

Self-training using CNG+SVM on character, lexical and syntactic views respectively: This gives three baselines. It self-trains two classifiers from the character 3-gram, lexical, and syntactic views using CNG and SVM classifiers (Kourtis and Stamatatos, 2011). CNG is a profile-based method which represents the author as the N most frequent character n -grams of all his/her training texts. The original method applied only CNG and SVM on the character n -gram view. Since our results show that its performance is extremely poor, we are curious what the reason is. Can this be due to the classifier or to the view? In order to differentiate the effects of views and classifiers, we present two additional types of baselines. The first type is to extend *CNG+SVM* method to lexical and syntactic views as well. The second type is to extend *CNG+SVM* method by replacing *CNG* with *LR* to show a fair comparison with our framework.

Self-training using LR+SVM on character, lexical, and syntactic views: This is the second type extension. It also gives us three baselines. It again uses the character, lexical and syntactic view and *SVM* as one of the two classifiers. The other classifier uses *LR* rather than *CNG*.

Co-training using LR on Char+Lex, Char+Syn, and Lex+Syn views: This also gives us three baselines. Each baseline co-trains two classifiers from every two views of the character 3-gram, lexical, and syntactic views.

4.3 Results and analysis

(1) Effects of learning algorithms

We first evaluate the effects of learning algorithms on tri-training. We use SVM and LR as the learners as they are among the best methods.

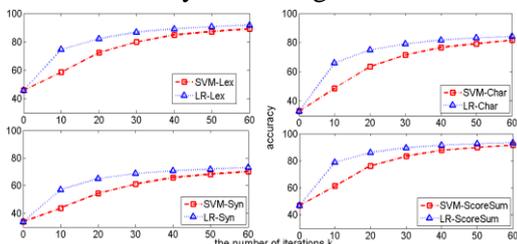


Figure 2. Effects of SVM and LR on tri-training

The effects of SVM and LR on tri-training are shown in Fig. 2. For the aggregation results, we draw the curves for ScoreSum. The results for other two strategies are similar. It is clear that LR outperforms SVM by a large margin for tri-training when the number of iterations (k) is

small. One possible reason is that LR is more tolerant to over-fitting caused by the small number of training samples. Hence, we use LR for tri-training in all experiments.

(2) Effects of aggregation strategies

We show the effects of the three proposed aggregation strategies. Table 1 indicates that ScoreSum (SS) is the best.

k	Single View Results			Aggregated Results		
	Lex	Char	Syn	SM	SS	SQ
0	45.75	32.88	33.96	41.11	46.85	44.61
10	74.63	66.05	56.99	73.41	78.82	76.41
20	82.30	74.92	65.05	81.63	86.19	84.05
30	86.86	79.12	68.85	85.29	89.69	87.74
40	89.16	81.81	70.85	87.83	91.52	89.99
50	90.56	83.14	72.06	89.11	92.58	91.17
60	91.69	84.13	73.23	90.05	93.15	91.82

Table 1. Effects of three aggregation strategies: ScoreMax(SM), ScoreSum(SS), and ScoreSq-Sum(SQ)

We also observe that both ScoreSum and ScoreSqSum (SQ) perform better than ScoreMax (SM) and all single view cases. This suggests that the decision made from a number of scores is much more reliable than that made from only one score. ScoreSum is our default strategy.

(3) Effects of data augmenting strategies

We now see the effects of data adding methods to augment the labeled set in Fig. 3.

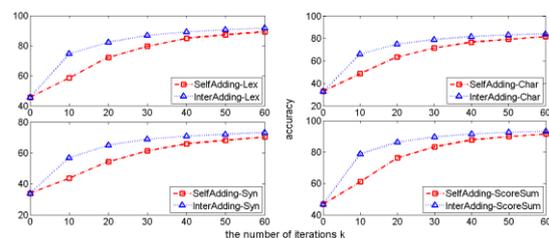


Figure 3. Effects of data augmenting methods on tri-training

We use two strategies. One is our *InterAdding* approach and the other is the *SelfAdding* approach in (Kourtis and Stamatatos, 2011), as introduced in Section 3.1. We can see that by adding newly classified samples by two classifiers to the third view, tri-training gets better and better results rapidly. For example, the accuracy for $k = 10$ iterations grows from 61.24 for SelfAdding to 78.82 for InterAdding, an absolute increase of 17.58%. This implies that by integrating more information from other views, learning can improve greatly.

(4) Comparison with self-training baselines

We show the results of CNG+SVM in Table 2. It is clear that CNG is almost unable to correctly

classify any test case. Its accuracy is only 1.26% at the start. This directly leads to the failure of the self-training. The reason is that the other classifier SVM can augment nearly 0 documents from the unlabeled set. We also tuned the parameter N for *CNG*, but it makes little difference.

k	Self-Training on Char		Aggregated Results		
	CNG	SVM	SM	SS	SQ
0	1.26	33.22	32.35	32.47	27.00
10	1.26	32.35	32.35	32.47	27.00
20	1.26	32.35	32.35	32.47	27.00
30	1.26	32.35	32.35	32.47	27.00
40	1.26	33.60	33.60	33.69	29.07
50	1.26	33.60	33.60	33.69	29.07
60	1.27	33.54	33.60	33.69	29.07

Table 2. Results for the *CNG+SVM* baseline

To distinguish the effects of views from classifiers, we conduct two more types of experiments. First, we apply *CNG+SVM* to the lexical and syntactic views. The results are even worse. Its accuracy drops to 0.58% and 1.21%, respectively. Next, we replace *CNG* with *LR* and apply *LR+SVM* to all three views. We only show their best results in Table 3, either on a single view or aggregation. The details are omitted due to space limitations. We can see significant improvements over their corresponding results of *CNG+SVM*. This demonstrates that the learning methods are critical to self-training as well.

k	Tri Train	SelfTrain:CNG+SVM			SelfTrain:LR+SVM		
		Char	lex	Syn	Char	Lex	Syn
0	46.85	33.22	45.44	34.50	33.22	45.75	34.48
10	78.82	32.47	45.44	34.50	62.56	73.78	51.94
20	86.19	32.47	45.44	34.09	71.21	81.44	59.88
30	89.69	32.47	45.44	34.09	75.21	84.68	63.70
40	91.52	33.69	45.44	34.09	77.46	88.25	65.74
50	92.58	33.69	45.44	34.09	78.64	88.25	67.45
60	93.15	33.69	45.44	34.09	79.54	89.31	68.37

Table 3. Self-training variations

From Table 3, we can also see that our tri-training approach outperforms all self-training baselines by a large margin. For example, the accuracy for *LR+SVM* on the lexical view is 89.31%. Although this is the best for self-training, it is worse than 93.15% of tri-training.

The reason that self-training does not work well in general is the following: When the training set is small, the available data may not reflect the true distribution of the whole data. Then classifiers will be biased and their classifications will be biased too. In testing, the biased classifiers will not have good accuracy. However, in tri-training, and co-training, each individual view may be biased but the views are independent. Then each view is more likely to produce random samples for the other views and thus reduce the bias of each view as the iterations progress.

(5) Comparison with co-training baselines

We now compare tri-training with co-training (Blum and Mitchell, 1998) in Table 4. Again, tri-training beats co-training consistently. The best performance of co-training is 92.81% achieved on the character and lexical views after 60 iterations. However, the accuracy is worse than that of tri-training. The key reason is that tri-training considers three views, while co-training uses only two. Also, the predictions by two classifiers are more reliable than those by one classifier.

k	Tri Train	Co-Train		
		Char+Lex	Char+Syn	Lex+Syn
0	46.85	45.75	42.02	45.75
10	78.82	78.84	75.89	78.85
20	86.19	86.02	82.59	85.63
30	89.69	89.32	85.77	88.98
40	91.52	91.14	87.52	91.16
50	92.58	92.19	88.46	92.02
60	93.15	92.81	89.21	92.50

Table 4. Co-training vs. tri-training

In (Qian, et al., 2014), we systematically investigated the effects of learning methods and views using a special co-training approach with two views. Learning was applied on two views but the data augmentation method was like that in self-training. The best result there was 91.23%, worse than 92.81% of co-training here in Table 4, which is worse than 93.15% of Tri-Training.

Overall, Tri-training performs the best and co-training is better than self-training and co-self-training. This indicates that learning on different views can better exploit the redundancy in texts to achieve superior classification results.

5 Conclusion

In this paper, we investigated the problem of authorship attribution with very few labeled examples. A novel three-view tri-training method was proposed to utilize natural views of human languages, i.e., the character, lexical and syntactic views, for classification. We evaluated the proposed method and compared it with state-of-the-art baselines. Results showed that the proposed method outperformed all baseline methods.

Our future work will extend the work by including more views such as the stylistic and vocabulary richness views. Additional experiments will also be conducted to determine the general behavior of the tri-training approach.

Acknowledgements

This work was supported in part by the NSFC projects (61272275, 61232002, 61379044), and the 111 project (B07037).

References

- S. Argamon, C. Whitelaw, P. Chase, S. R. Hota, N. Garg, and S. Levitan. 2007. Stylistic text classification using functional lexical features. *JASIST* 58, 802–822
- A. Blum and T. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In: *COLT*. pp. 92–100
- J. Burrows. 1992. Not unless you ask nicely: The interpretative nexus between analysis and information. *Literary and Linguistic Computing* 7:91-109.
- J. Burrows. 2007. All the way through: Testing for authorship in different frequency data. *LLC* 22, 27–47
- R-E. Fan, K-W. Chang, C-J. Hsieh, X-R. Wang, and C-J. Lin. 2008. Liblinear: A library for large linear classification. *JMLR* 9, 1871–1874
- J. Diederich, J. Kindermann, E. Leopold, G. Paass, G. F. Informationstechnik, and D-S. Augustin. 2000. Authorship attribution with support vector machines. *Applied Intelligence* 19:109-123.
- M. Gamon. 2004. Linguistic correlates of style: authorship classification with deep linguistic analysis features. In *COLING*.
- N. Graham, G. Hirst, and B. Marthi. 2005. Segmenting documents by stylistic character. *Natural Language Engineering*, 11:397-415.
- J. Grieve. 2007. Quantitative authorship attribution: An evaluation of techniques. *LLC* 22:251-270.
- H. van Halteren, F. Tweedie, and H. Baayen. 1996. Outside the cave of shadows: using syntactic annotation to enhance authorship attribution. *Literary and Linguistic Computing* 11:121-132.
- H. van Halteren. 2007. Author verification by linguistic profiling: An exploration of the parameter space. *TSLP* 4, 1–17
- G. Hirst, and O. Feiguina. 2007. Bigrams of syntactic labels for authorship discrimination of short texts. *LLC* 22, 405–417
- N. Jindal and B. Liu. 2008. Opinion spam and analysis. In: *WSDM*. pp. 29–230
- T. Joachims. 2007. www.cs.cornell.edu/people/tj/svmmlight/old/svmmulticlassv2.12.html
- S. Kim, H. Kim, T. Weneringer, J. Han, and H. D. Kim. 2011. Authorship classification: a discriminative syntactic tree mining approach. In: *SIGIR*. pp. 455–464
- D. Klein and C. D. Manning. 2003. Accurate unlexicalized parsing. In: *ACL*. pp. 423–430
- I. Kourtis and E. Stamatatos, 2011. Author identification using semi-supervised learning. In: *Notebook for PAN at CLEF 2011*
- J. Li, R. Zheng, and H. Chen. 2006. From fingerprint to writeprint. *Communications of the ACM* 49:76-82.
- K. Luyckx and W. Daelemans, 2008. Authorship attribution and verification with many authors and limited data. In: *COLING*. pp. 513–520
- D. Madigan, A. Genkin, D. Lewis, A. Argamon, D. Fradkin, and L. Ye, 2005. Author Identification on the Large Scale. In *CSNA*.
- K. Nigam and R. Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *Proc. of CIKM*, pp.86–93
- T. Qian, B. Liu. 2013 Identifying Multiple Userids of the Same Author. *EMNLP*, pp. 1124-1135
- T. Qian, B. Liu, M. Zhong, G. He. 2014. Co-Training on Authorship Attribution with Very Few Labeled Examples: Methods. vs. Views. In *SIGIR*, to appear.
- R. Schwartz, O. Tsur, A. Rappoport, M. Koppel. 2013. Authorship Attribution of Micro-Messages. *EMNLP*. pp. 1880-1891
- Y. Seroussi, F. Bohnert and Zukerman, 2012. Authorship attribution with author-aware topic models. In: *ACL*. pp. 264–269
- Y. Seroussi, I. Zukerman, and F. Bohnert. 2010. Collaborative inference of sentiments from texts. In: *UMAP*. pp. 195–206
- E. Stamatatos. 2007. Author identification using imbalanced and limited training texts. In: *TIR*. pp. 237–241
- E. Stamatatos. 2009. A survey of modern authorship attribution methods. *JASIST* 60:538–556
- Ö. Uzuner and B. Katz. 2005. A comparative study of language models for book and author recognition. *Proc. of the 2nd IJCNLP*, 969-980.

- Y. Zhao and J. Zobel. 2005. Effective and scalable authorship attribution using function words. *In Proc. of Information Retrieval Technology*, 174-189.
- R. Zheng, J. Li, H. Chen, and Z. Huang. 2006. A framework for authorship identification of online messages: Writing style features and classification techniques. *JASIST* 57:378-393.

Automation and Evaluation of the Keyword Method for Second Language Learning

Gözde Özbal
Trento RISE
Trento, Italy
gozbalde@gmail.com

Daniele Pighin
Google
Zürich, Switzerland
biondo@google.com

Carlo Strapparava
FBK-irst
Trento, Italy
strappa@fbk.eu

Abstract

In this paper, we combine existing NLP techniques with minimal supervision to build memory tips according to the keyword method, a well established mnemonic device for second language learning. We present what we believe to be the first extrinsic evaluation of a creative sentence generator on a vocabulary learning task. The results demonstrate that NLP techniques can effectively support the development of resources for second language learning.

1 Introduction

The keyword method is a mnemonic device (Cohen, 1987; Thompson, 1987) that is especially suitable for vocabulary acquisition in second language learning (Mizumoto and Kansai, 2009; Hummel, 2010; Shen, 2010; Tavakoli and Gerami, 2013). In this method, a *target* word in a foreign language L2 can be learned by a native speaker of another language L1 in two main steps: 1) one or more L1 words, possibly referring to a concrete entity, are chosen based on orthographic or phonetic similarity with the target word; 2) an L1 sentence is constructed in which an association between the translation of the target word and the keyword(s) is established, so that the learner, when seeing or hearing the word, immediately recalls the keyword(s). To illustrate, for teaching the Italian word *cuore* which means *heart* in English, the learner might be asked to imagine “*a lonely heart with a hard core*”.

The keyword method has already been proven to be a valuable teaching device. However, the preparation of the memorization tips for each new word is an activity that requires considerable time, linguistic competence and creativity. To the best of our knowledge, there is only one study which attempts to automate the mechanism of the keyword method. In (Özbal and Strapparava, 2011),

we proposed to automate the keyword method by retrieving sentences from the Web. However, we did not provide any evaluation to demonstrate the effectiveness of our approach in a real life scenario. In addition, we observed that retrieval poses severe limitations in terms of recall and sentence quality, and it might incur copyright violations.

In this paper, we overcome these limitations by introducing a semi-automatic system implementing the keyword method that builds upon the keyword selection mechanism of Özbal and Strapparava (2011) and combines it with a state-of-the-art creative sentence generation framework (Özbal et al., 2013). We set up an experiment to simulate the situation in which a teacher needs to prepare material for a vocabulary teaching resource. According to our scenario, the teacher relies on automatic techniques to generate relatively few, high quality mnemonics in English to teach Italian vocabulary. She only applies a very light supervision in the last step of the process, in which the most suitable among the generated sentences are selected before being presented to the learners. In this stage, the teacher may want to consider factors which are not yet in reach of automatic linguistic processors, such as the evocativeness or the memorability of a sentence. We show that the automatically generated sentences help learners to establish memorable connections which augment their ability to assimilate new vocabulary. To the best of our knowledge, this work is the first documented extrinsic evaluation of a creative sentence generator on a real-world application.

2 Related work

The effectiveness of the keyword method (KM) is a well-established fact (Sarıçoban and Başıbek, 2012). Sommer and Gruneberg (2002) found that using KM to teach French made learning easier and faster than conventional methods. Sagarra and Alba (2006) compared the effectiveness of

three learning methods including the semantic mapping, rote memorization (i.e., memorization by pure repetition, with no mnemonic aid) and keyword on beginner learners of a second language. Their results show that using KM leads to better learning of second language vocabulary for beginners. Similar results have been reported by Sarıçoban and Başibek (2012) and Tavakoli and Gerami (2013). Besides all the experimental results demonstrating the effectiveness of KM, it is worthwhile to mention about the computational efforts to automate the mechanism. In (Özbal and Strapparava, 2011) we proposed an automatic vocabulary teaching system which combines NLP and IR techniques to automatically generate memory tips for vocabulary acquisition. The system exploits orthographic and phonetic similarity metrics to find the best L2 keywords for each target L1 word. Sentences containing the keywords and the translation of the target word are retrieved from the Web, but we did not carry out an evaluation of the quality or the coverage of the retrieved sentences. In Özbal et al. (2013) we proposed an extensible framework for the generation of creative sentences in which users are able to force several words to appear in the sentences. While we had discussed the potentiality of creative sentence generation as a useful teaching device, we had not validated our claim experimentally yet. As a previous attempt at using NLP for education, Manurung et al. (2008) employ a riddle generator to create a language playground for children with complex communication needs.

3 Memory tip generation

Preparing memory tips based on KM includes two main ingredients: one or more keywords which are orthographically or phonetically similar to the L2 word to be learned; and a sentence in which the keywords and the translation of the target L2 word are combined in a meaningful way. In this section, we detail the process that we employed to generate such memory tips semi-automatically.

3.1 Target word selection and keyword generation

We started by compiling a collection of Italian nouns consisting of three syllables from various resources for vocabulary teaching including <http://didattica.org/italiano.htm> and <http://ielanguages.com>, and produced a list of 185 target L2 words. To gen-

erate the L1 keywords for each target word, we adopted a similar strategy to Özbal and Strapparava (2011). For each L2 target word t , the keyword selection module generates a list of possible keyword pairs, K . A keyword pair $k \in K$ can either consist of two non-empty strings, i.e., $k = [w_0, w_1]$, or of one non-empty and one empty string, i.e., $w_1 = \epsilon$. Each keyword pair has the property that the concatenation of its elements is either orthographically or phonetically similar to the target word t . Orthographic and phonetic similarity are evaluated by means of the Levenshtein distance (Levenshtein, 1966). For orthographic similarity, the distance is calculated over the characters in the words, while for phonetic similarity it is calculated over the phonetic representations of t and $w_0 + w_1$. We use the CMU pronunciation dictionary¹ to retrieve the phonetic representation of English words. For Italian words, instead, their phonetic representation is obtained from an unpublished phonetic lexicon developed at FBK-irst.

3.2 Keyword filtering and ranking

Unlike in (Özbal and Strapparava, 2011), where we did not enforce any constraints for selecting the keywords, in this case we applied a more sophisticated filtering and ranking strategy. We require at least one keyword in each pair to be a content word; then, we require that at least one keyword has length ≥ 3 ; finally, we discard pairs containing at least one proper noun. We allowed the keyword generation module to consider all the entries in the CMU dictionary, and rank the keyword pairs based on the following criteria in decreasing order of precedence: 1) Keywords with a smaller orthographic/phonetic distance are preferred; 2) Keywords consisting of a single word are preferred over two words (e.g., for the target word *lavagna*, which means *blackboard*, *lasagna* takes precedence over *love* and *onion*); 3) Keywords that do not contain stop words are preferred (e.g., for the target word *pettine*, which means *comb*, the keyword pair *pet* and *inn* is ranked higher than *pet* and *in*, since *in* is a stop word); 4) Keyword pairs obtained with orthographic similarity are preferred over those obtained with phonetic similarity, as learners might be unfamiliar with the phonetic rules of the target language. For example, for the target word *forbice*, which means *scissors*,

¹<http://www.speech.cs.cmu.edu/cgi-bin/cmudict>

Group	Target	Sentence
A1	campagna	a <i>company</i> runs the country
A1	isola	an <i>island</i> of remote isolated communities
A1	fabbrica	a fabric worker in a factory
A1	bagnino	lifeguards carry <i>no bag</i>
A1	inverno	the <i>inferno</i> started, winter left
A1	cielo	the sky has no <i>ceiling</i>
A1	marrone	blood and <i>marrow</i> in a brown water
A1	cuore	the lonely heart has hard <i>core</i>
A1	coperta	a piece of <i>copper</i> in the corner of a blanket
A1	locanda	an inn oak door with <i>lock and key</i>
A2	piazza	a square building serves a free <i>pizza</i>
A2	calzino	big bloke with sock in the <i>casino</i>
A2	scatola	a cardboard box sat in a <i>scuttle</i> of a house
A2	ragazzo	boys also have <i>rag</i> dolls
A2	angolo	a corner kick came at an <i>angle</i>
A2	cestino	a <i>teen</i> movie uses basket to play the <i>chess</i>
A2	carbone	the coal is the form of <i>carbon</i>
A2	cassetto	a blank <i>cassette</i> tape is in a drawer
A2	farfalla	the butterflies are <i>far</i> in the <i>fall</i>
A2	tovaglia	a damp cloth <i>towel</i>
B1	duomo	the old cathedral has a <i>dome</i>
B1	aceto	a vinegar sauce contains the <i>acid</i>
B1	nuvola	the sophisticated <i>novel</i> depicts the cloud
B1	chiesa	the Catholic church has Swiss <i>cheese</i>
B1	bacino	the explosion <i>in the back</i> broke the <i>pelvis</i>
B1	maiale	a pork meat comes in the <i>mail</i>
B1	minestra	Chinese <i>ministries</i> have soup
B1	estate	this <i>estate</i> is for summer
B1	bozzolo	a <i>buzz</i> comes wrapped in the cocoon
B1	arnese	<i>harness</i> a technology to develop a tool
B2	asino	an <i>Asian</i> elephant is riding a donkey
B2	miele	do not make honey to walk a <i>mile</i>
B2	polmone	crowded <i>pullmans</i> stop the lungs
B2	fagiolo	a topical <i>facial</i> bean cream
B2	fiore	a <i>fire</i> in a flower market
B2	compressa	the clay tablet is in the <i>compressed</i> form
B2	cavallo	horse running fast in <i>cavalry</i>
B2	fiume	the muddy river has smoke and <i>fumes</i>
B2	pittore	a famous painter has precious <i>pictures</i>
B2	manico	<i>manic</i> people have broken necks

Table 1: Sentences used in the vocabulary acquisition experiment.

the keyword pair *for* and *bid* is preferred to *for* and *beach*.

We selected up to three of the highest ranked keyword pairs for each target word, obtaining 407 keyword combinations for the initial 185 Italian words, which we used as the input for the sentence generator.

3.3 Sentence generation

In this step, our goal was to generate, for each Italian word, sentences containing its L1 translation and the set of orthographically (or phonetically) similar keywords that we previously selected. For each keyword combination, starting from the top-ranked ones, we generated up to 10 sentences by allowing any known part-of-speech for the keywords. The sentences were produced by the state

of the art sentence generator of Özbal et al. (2013). The system relies on two corpora of automatic parses as a repository of sentence templates and lexical statistics. As for the former, we combined two resources: a corpus of 16,000 proverbs (Mihalcea and Strapparava, 2006) and a collection of 5,000 image captions² collected by Rashtchian et al. (2010). We chose these two collections since they offer a combination of catchy or simple sentences that we expect to be especially suitable for second language learning. As for the second corpus, we used LDC’s English GigaWord 5th Edition³. Of the 12 feature functions described in (Özbal et al., 2013), we only implemented the following scorers: Variety (to prevent duplicate words from appearing in the sentences); Semantic Cohesion (to enforce the generation of sentence as lexically related to the target words as possible); Alliteration, Rhyme and Plosive (to introduce hooks to echoic memory in the output); Dependency Operator and *N*-gram (to enforce output grammaticality).

We observed that the sentence generation module was not able to generate a sentence for 24% of the input configurations. For comparison, when we attempted to retrieve sentences from the Web as suggested in Özbal and Strapparava (2011), we could collect an output for less than 10% of the input configurations. Besides, many of the retrieved sentences were exceedingly long and complex to be used in a second language learning experiment.

3.4 Sentence selection

For each L1 keyword pair obtained for each L2 target word, we allowed the system to output up to 10 sentences. We manually assessed the quality of the generated sentences in terms of meaningfulness, evocativeness and grammaticality to select the most appropriate sentences to be used for the task. In addition, for keyword pairs not containing the empty string, we prioritized the sentences in which the keywords were closer to each other. For example, let us assume that we have the keywords *call* and *in* for the target word *collina*. Among the sentences “*The girl received a call in the bathroom*” and “*Call the blond girl in case you need*”, the first one is preferred, since the keywords are closer to each other. Furthermore, we gave priority to the sentences that included the keywords

²<http://vision.cs.uiuc.edu/pascal-sentences/>

³<http://www ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2011T07>

in the right order. To illustrate, for the same keywords and the target words, we would prefer the sentence “*I called him in the morning yesterday*” over “*You talk a lot in a call*”.

Accordingly, for each target word in random order, we sequentially scanned the outputs generated for each keyword pair. As soon as a sentence of adequate quality was found, we added it to our evaluation data and moved on to the next keyword. We continued this process until we selected a sentence for 40 distinct target words, which we set as the target size of the experiment. We had to inspect the outputs generated for 48 target words before we were able to select 40 good examples, meaning that for 17% of the target words the sentence generator could not produce a sentence of acceptable quality.

4 Experiment setup

For our experiment, we drew inspiration from Sagarra and Alba (2006). We compared the retention error rate of learners who tried to memorize new words with or without the aid of the automatically generated sentences. Through academic channels, we recruited 20 native English speakers with no prior knowledge of Italian.⁴

After obtaining the sentences as explained in Section 3, we shuffled and then divided the whole set including 40 target words together with their translation, the generated keywords and sentences into 2 batches (A, B) and further divided each batch into 2 groups consisting of 10 elements (A1, A2, B1 and B2). The set of sentences assigned to each group is listed in Table 1: Column “*Target*” reports the Italian target word being taught; Column “*Sentence*” shows the automatically generated sentence, where the translation of the target word is shown in bold and the keyword(s) in italic. For the experiments, we randomly assigned each subject to one of the batches (A or B). Then, each subject was asked to memorize all the word pairs in a batch, but they would see the memory tips only for one of the two groups, which was again randomly assigned. This approach resulted in 4 different memorization exercises, namely 1) A1 with tips and A2 without, 2) A2 with tips and A1 without, 3) B1 with tips and B2 without, 4) B2 with tips and B1 without.

⁴We preferred to select the experiment subjects in person as opposed to crowdsourcing the evaluation to be able to verify the proficiency of the subjects in the two languages and to ensure the reliability of the outcome of the evaluation.

Group	Error rate (%)		Reduction	
	Rote	KW	Δ_e	$\%_e$
A1	4.08	3.39	0.69	16.95
A2	12.07	10.42	1.65	13.69
B1	12.77	10.00	2.77	21.67
B2	22.50	12.50	10.00	44.44
Macro-average	12.85	9.08	3.78	29.39
Micro-average	11.27	8.25	3.02	26.76

Table 2: Per-group and overall retention error rate when using rote or keyword-aided (KW) memorization.

When memorizing the translations without the aid of memory tips, the subjects were instructed to focus only on the Italian word and its English translation and to repeat them over and over in their mind. Conversely, when relying on the automatic memory tips the subjects were shown the word, its translation and the generated sentence including the keywords. In this case, the subjects were instructed to read the sentence over and over trying to visualize it.

After going through each set of slides, we distracted the subjects with a short video in order to reset their short term memory. After that, their retention was tested. For each Italian word in the exercise, they were asked to select the English translation among 5 alternatives, including the correct translation and 4 other words randomly selected from the same group. In this way, the subjects would always have to choose among the words that they encountered during the exercise.⁵ We also added an extra option “*I already knew this word*” that the subjects were instructed to select in case they already knew the Italian word prior to taking part in the experiment.

5 Experiment results

Table 2 summarizes the outcome of the experiment. The contribution of the automatically generated sentences to the learning task is assessed in terms of error rate-reduction, which we measure both within each group (rows 1-4) and on the whole evaluation set (rows 5-6). Due to the presence of the “*I already knew this word*” option in the learning-assessment questionnaire, the number of the actual answers provided by each subject can be slightly different, hence the difference between macro- and micro-average.

⁵Otherwise, they could easily filter out the wrong answers just because they were not exposed to them recently.

The error rate for each memorization technique t (where $t = R$ for “Rote memorization” and $t = K$ for “keyword-aided memorization”) is calculated as: $e_t = \frac{i_t}{c_t + i_t}$, where c_t and i_t are the number of correct and incorrect answers provided by the subjects, respectively. The absolute error rate reduction Δe is calculated as the absolute difference in error rate between rote and keyword-aided memorization, i.e.: $\Delta e = e_R - e_K$. Finally, the relative error rate reduction $\%_e$ is calculated as the ratio between the absolute error rate reduction Δe and the error rate of rote memorization e_R , i.e.: $\%_e = \frac{\Delta e}{e_R} = \frac{e_R - e_K}{e_R}$.

The overall results (rows 5 and 6 in Table 2) show that vocabulary learning noticeably improves when supported by the generated sentences, with error rates dropping by almost 30% in terms of macro-average (almost 27% for micro-average). The breakdown of the error rate across the 4 groups shows a clear pattern. The results clearly indicate that one group (A1) by chance contained easier words to memorize as shown by the low error rate (between 3% and 4%) obtained with both methods. Similarly, groups A2 and B1 are of average difficulty, whereas group B2 appears to be the most difficult, with an error rate higher than 22% when using only rote memorization. Interestingly, there is a strong correlation (Pearson’s $r = 0.85$) between the difficulty of the words in each group (measured as the error rate on rote memorization) and the positive contribution of the generated sentences to the learning process. In fact, we can see how the relative error rate reduction $\%_e$ increases from $\sim 17\%$ (group A1) to almost 45% (group B2). Based on the results obtained by Sagarra and Alba (2006), who showed that the keyword method results in better long-term word retention than rote memorization, we would expect the error rate reduction to be even higher in a delayed post-test. All in all, these findings clearly support the claim that a state-of-the-art sentence generator can be successfully employed to support keyword-based second language learning. After completing their exercise, the subjects were asked to provide feedback about their experience as learners. We set up a 4-items Likert scale (Likert, 1932) where each item consisted of a statement and a 5-point scale of values ranging from (1) [I strongly disagree] to (5) [I strongly agree]. The distribution of the answers to the questions is shown in Table 3. 60% of the subjects acknowledged that the memory tips helped them in

Question	Rating (%)				
	1	2	3	4	5
Sentences helped	5	20	15	35	25
Sentences are grammatical	-	25	30	35	10
Sentences are catchy	-	25	10	50	15
Sentences are witty	-	25	25	50	-

Table 3: Evaluation of the generated sentences on a 5-point Likert scale.

the memorization process; 45% found that the sentences were overall correct; 65% confirmed that the sentences were catchy and easy to remember; and 50% found the sentences to be overall witty although the sentence generator does not include a mechanism to generate humor. Finally, it is worth mentioning that none of the subjects noticed that the sentences were machine generated, which we regard as a very positive assessment of the quality of the sentence generation framework. From their comments, it emerges that the subjects actually believed that they were just comparing two memorization techniques.

6 Conclusion and Future Work

In this paper, we have presented a semi-automatic system for the automation of the keyword method and used it to teach 40 Italian words to 20 English native speakers. We let the system select appropriate keywords and generate sentences automatically. For each Italian word, we selected the most suitable among the 10 highest ranked suggestions and used it for the evaluation. The significant reduction in retention error rate (between 17% and 45% on different word groups) for the words learned with the aid of the automatically generated sentences shows that they are a viable low-effort alternative to human-constructed examples for vocabulary teaching.

As future work, it would be interesting to involve learners in an interactive evaluation to understand the extent to which learners can benefit from *ad-hoc* personalization. Furthermore, it should be possible to use frameworks similar to the one that we presented to automate other teaching devices based on sentences conforming to specific requirements (Dehn, 2011), such as *verbal chaining* and *acrostic*.

Acknowledgements

This work was partially supported by the PerTe project (Trento RISE).

References

- Andrew D. Cohen. 1987. The use of verbal and imagery mnemonics in second-language vocabulary learning. *Studies in Second Language Acquisition*, 9:43–61, 2.
- M.J. Dehn. 2011. *Working Memory and Academic Learning: Assessment and Intervention*. Wiley.
- K. M. Hummel. 2010. Translation and short-term L2 vocabulary retention: Hindrance or help? *Language Teaching Research*, 14(1):61–74.
- V. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710.
- R. Likert. 1932. A technique for the measurement of attitudes. *Archives of Psychology*, 22(140):1–55.
- Ruli Manurung, Graeme Ritchie, Helen Pain, Annalu Waller, Dave O'Mara, and Rolf Black. 2008. The Construction of a Pun Generator for Language Skill Development. *Appl. Artif. Intell.*, 22(9):841–869, October.
- R. Mihalcea and C. Strapparava. 2006. Learning to laugh (automatically): Computational models for humor recognition. *Journal of Computational Intelligence*, 22(2):126–142, May.
- A. Mizumoto and O. T. Kansai. 2009. Examining the effectiveness of explicit instruction of vocabulary learning strategies with Japanese EFL university students. *Language Teaching Research* 13, 4.
- Gözde Özbal and Carlo Strapparava. 2011. MEANS: Moving Effective Assonances for Novice Students. In *Proceedings of the 16th International Conference on Intelligent User Interfaces (IUI 2011)*, pages 449–450, New York, NY, USA. ACM.
- Gözde Özbal, Daniele Pighin, and Carlo Strapparava. 2013. BRAINSUP: Brainstorming Support for Creative Sentence Generation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 1446–1455, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. 2010. Collecting image annotations using amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, CSLDAMT '10, pages 139–147, Stroudsburg, PA, USA. Association for Computational Linguistics.
- N. Sagarra and M. Alba. 2006. The key is in the keyword: L2 vocabulary learning methods with beginning learners of spanish. *The Modern Language Journal*, 90(2):228–243.
- A. Sariçoban and N. Başıbek. 2012. Mnemonics technique versus context method in teaching vocabulary at upper-intermediate level. *Journal of Education and Science*, 37(164):251–266.
- Helen H. Shen. 2010. Imagery and verbal coding approaches in Chinese vocabulary instruction. *Language Teaching Research*, 14(4):485–499.
- Steffen Sommer and Michael Gruneberg. 2002. The use of linkword language computer courses in a classroom situation: a case study at rugby school. *Language Learning Journal*, 26(1):48–53.
- M. Tavakoli and E. Gerami. 2013. The effect of keyword and pictorial methods on EFL learners' vocabulary learning and retention. *PORTA LINGUARUM*, 19:299–316.
- G. Thompson. 1987. Using bilingual dictionaries. *ELT Journal*, 41(4):282–286. cited By (since 1996)6.

Citation Resolution: A method for evaluating context-based citation recommendation systems

Daniel Duma

University of Edinburgh

D.C.Duma@sms.ed.ac.uk

Ewan Klein

University of Edinburgh

ewan@staffmail.ed.ac.uk

Abstract

Wouldn't it be helpful if your text editor automatically suggested papers that are relevant to your research? Wouldn't it be even better if those suggestions were contextually relevant? In this paper we name a system that would accomplish this a *context-based citation recommendation (CBCR) system*. We specifically present Citation Resolution, a method for the evaluation of CBCR systems which exclusively uses readily-available scientific articles. Exploiting the human judgements that are already implicit in available resources, we avoid purpose-specific annotation. We apply this evaluation to three sets of methods for representing a document, based on a) the contents of the document, b) the surrounding contexts of citations to the document found in other documents, and c) a mixture of the two.

1 Introduction

Imagine that you were working on a draft paper which contained a sentence like the following:¹

A variety of coherence theories have been developed over the years ... and their principles have found application in many symbolic text generation systems (e.g. [CITATION HERE])

Wouldn't it be helpful if your editor automatically suggested some references that you could cite here? This is what a citation recommendation system ought to do. If the system is able to take into account the *context* in which the citation occurs — for example, that papers relevant to our example above are not only about text generation

¹Adapted from the introduction to Barzilay and Lapata (2008)

systems, but specifically mention applying coherence theories — then this would be much more informative. So we define a *context-based* citation recommendation (CBCR) system as one that assists the author of a draft document by suggesting other documents with content that is relevant to a particular context in the draft.

Our longer term research goal is to provide suggestions that satisfy the requirements of specific expository or rhetorical tasks, e.g. provide support for a particular argument, acknowledge previous work that uses the same methodology, or exemplify work that would benefit from the outcomes of the author's work. However, our current paper has more modest aims: we present initial results using existing IR-based approaches and we introduce an evaluation method and metric. CBCR systems are not yet widely available, but a number of experiments have been carried out that may pave the way for their popularisation, e.g. He et al. (2010), Schäfer and Kasterka (2010) and He et al. (2012). It is within this early wave of experiments that our work is framed.

A main problem we face is that evaluating the performance of these systems ultimately requires human judgement. This can be captured as a set of relevance judgements for candidate citations over a corpus of documents, which is an arduous effort that requires considerable manual input and very careful preparation. In designing a context-based citation recommendation system, we would ideally like to minimise these costs.

Fortunately there is already an abundance of data that meets our requirements: every scientific paper contains human “judgements” in the form of citations to other papers which are contextually appropriate: that is, relevant to specific passages of the document and aligned with its argumentative structure. Citation Resolution is a method for evaluating CBCR systems that is exclusively based on this source of human judgements.

Let's define some terminology. In the following passage, the strings 'Scott and de Souza, 1990' and 'Kibble and Power, 2004' are both *citation tokens*:

A variety of coherence theories have been developed over the years ... and their principles have found application in many symbolic text generation systems (e.g. Scott and de Souza, 1990; Kibble and Power, 2004)

Note that a citation token can use any standard format. Furthermore

- a *citation context* is the context in which a citation token occurs, with no limit as to representation of this context, length or processing involved;
- a *collection-internal reference* is a reference in the bibliography of the source document that matches a document in a given corpus;
- a *resolvable citation* is an in-text citation token which resolves to a collection-internal reference.

2 Related work

While the existing work in this specific area is far from extensive, previous experiments in evaluating context-based citation recommendation systems have used one of three approaches. First, evaluation can be carried out through user studies, which is costly because it cannot be reused (e.g. Chandrasekaran et al. (2008)).

Second, a set of relevance judgements can be created for repeated testing. Ritchie (2009) details the building of a large set of relevance judgements in order to evaluate an experimental document retrieval system. The judgements were mainly provided by the authors of papers submitted to a locally organised conference, for over 140 queries, each of them being the main research question of one paper. This is a standard approach in IR, known as building a *test collection* (Sanderson, 2010), which the author herself notes was an arduous and time-consuming task.

Third, as we outlined above, existing citations between papers can be exploited as a source of human judgements. The most relevant previous work on this is He et al. (2010), who built an experimental CBCR system using the whole index of CiteSeerX as a test collection (over 450,000 documents). They avoided direct human evaluation and instead used three relevance metrics:

- *Recall*, the presence of the original reference in the list of suggestions generated by the system;
- *Co-cited probability*, a ratio between, on the one hand, the number of papers citing both the original reference and a recommended one, and on the other hand, the number of papers citing either of them; and
- *Normalized Discounted Cumulative Gain*, a measure based on the rank of the original reference in the list of suggested references, its score decreasing logarithmically.

However, these metrics fail to adequately recognise that the particular reference used by an author e.g. in support of an argument or as exemplification of an approach, may not be the most appropriate that could be found in the whole collection. This does not just amount to a difference of opinion between different authors; it is possible that within a large enough collection there exists a paper which the original author herself would consider to be more appropriate by any criteria (persuasive power, discoverability or the publication, etc.) than the one actually cited in the paper. Also, given that recommending the original citation used by the author in first position is our key criterion, a metric with smooth discounting like NDCG is too lenient for our purposes.

We have then chosen *top-1 accuracy* as our metric, where every time the original citation is first on the list of suggestions, it receives a score of 1, and 0 otherwise, and these scores are averaged over all resolved citations in the document collection. This metric is intuitive in measuring the efficiency of the system at this task, as it is immediately interpretable as a percentage of success.

While previous experiments in CBCR, like the ones we have just presented, have treated the task as an Information Retrieval problem, our ultimate purpose is different and travels beyond IR into Question Answering. We want to ultimately be able to assess the reason a document was cited in the context of the argumentation structure of the document, following previous work on the automatic classification of citation function by Teufel et al. (2006), Liakata et al. (2012) and Schäfer and Kasterka (2010). We expect this will allow us to identify claims made in a draft paper and match them with related claims made in other papers for support or contrast, and so offer answers in the form of relevant passages extracted from the sug-

gested documents.

It is frequently observed that the reasons for citing a paper go beyond its contribution to the field and its relevance to the research being reported (Hyland, 2009). There is a large body of research on the motivations behind citing documents (MacRoberts and MacRoberts, 1996), and it is likely that this will come to play a part in our research in the future.

In this paper, however, we present our initial results which compare three different sets of IR-based approaches to generating the document representation for a CBCR system. One is based on the contents of the document itself, one is based on the existing contexts of citations of this paper in other documents, and the third is a mixture of the two.

3 The task: Citation Resolution

In this section we present the evaluation method in more abstract terms; for the implementation used in this paper, please see Sections 4 and 5. The core criterion of this task is to use only the human judgements that we have clearest evidence for. Let d be a document and R the collection of all documents that are referenced in d . We believe it is reasonable to assume that the author of document d knows enough about the contents of each document R_i to choose the most appropriate citation from the collection R for every citation context in the document.

This captures a very strong relevance judgement about the relation between a particular citation context in the document and a particular cited reference document. We use these judgements for evaluation: our task is to match every citation context in the document (i.e. the surrounding context of a citation token) with the right reference from the list of references cited by that paper.

This task differs somewhat from standard Information Retrieval, in that we are not trying to retrieve a document from a larger collection outside the source document, but trying to resolve the correct reference for a given citation context from an existing list of documents, that is, from the bibliography that has been manually curated by the authors. Our document collection used for retrieval is further composed of only the references of that document that we can access.

The algorithm for the task is presented in Figure 1. For any given *test document* (2), we first extract

all the citation tokens found in the text that correspond to a collection-internal reference (a). We then create a *document representation* of the referenced document (currently a Vector Space Model, but liable to change). This representation can be based on any information found in the document collection, excluding the document d itself: e.g. the text of the referenced document and the text of documents that cite it.

For each citation token we then extract its context (b.i), which becomes the *query* in IR terms. One way of doing this that we present here is to select a list of word tokens around the citation. We then attempt to *resolve* the citation by computing a score for the match between each reference representation and the citation context (b.ii). We rank all collection-internal references by this score in decreasing order, aiming for the original reference to be in the first position (b.iii).

In the case where multiple citations share the same context, that is, they are made in direct succession (e.g. “...*compared with previous approaches (Author (2005), Author and Author (2007))*”), the first n elements of the list of suggested documents all count as the first element. That is, if any of the references in a multiple citation of n elements appears in the first n positions of the list of suggestions, it counts as a successful resolution and receives a score of 1. The final score is averaged over all citation contexts processed.

The set of experiments we present here apply this evaluation to test a number of IR techniques which we detail in the next section.

1. Given document collection D
2. For every test document d
 - (a) For every reference r in its bibliography R
 - i. If r is in document collection D
 - ii. Add all inline citations C_r in d to list C
 - (b) For each citation c in C
 - i. Extract context ctx_c of c
 - ii. Choose which document r in R best matches ctx_c
 - iii. Measure accuracy

Figure 1: Algorithm for citation resolution.

4 Experiments

Our test corpus consists of approx. 9000 papers from the ACL Anthology² converted from PDF to

²<http://aclweb.org/anthology/>

XML format. This corpus, the rationale behind its selection and the process used to convert the files is described in depth in Ritchie et al. (2006). This is an ideal corpus for these tests for a large number of reasons, but these are key for us: all the papers are freely available, the ratio of collection-internal references for each paper is high (the authors measure it at 0.33) and it is a familiar domain for us.

For our tests, we selected the documents of this corpus with at least 8 collection-internal references. This yielded a total of 278 test documents and a total of 5446 resolvable citations.

We substitute all citations in the text with citation token placeholders and extract the citation context for each using a simple *window* of up to w words left and w words right around the placeholder. This produces a list of word tokens that is equivalent to a *query* in IR.

This is a frequently employed technique (He et al., 2010), although it is often observed that this may be too simplistic a method (Ritchie, 2009). Other methods have been tried, e.g. full sentence extraction (He et al., 2012) and comparing these methods is something we plan to incorporate in future work.

We then make the document’s collection-internal references our test collection D and use a number of methods for generating the document representation. We use the well-known Vector Space Model and a standard implementation of *tf-idf* and *cosine similarity* as implemented by the *scikit-learn* Python framework³. At present, we are applying no cut-off and just rank all of the document’s collection-internal references for each citation context, aiming to rank the correct one in the first positions in the list.

We tested three different approaches to generating a document’s VSM representation: *internal representations*, which are based on the contents of the document, *external representations*, which are built using a document’s incoming link citation contexts (following Ritchie (2009) and He et al. (2010)) and mixed representations, which are an attempt to combine the two.

- The internal representations of the documents were generated using three different methods: title plus abstract, full text and *passage*. *Passage* consists in splitting the document into half-overlapping passages of a fixed length of k words and choosing for each document the

passage with the maximum cosine similarity score with the query. We present the results of using 250, 300 and 350 as values for k .

- The external representations (*inlink_context*) are based on extracting the context around citation tokens to the document from other documents in the collection, excluding the set of test papers. This is the same as using the *anchor text* of a hyperlink to improve results in web-based IR (see Davison (2000) for extensive analysis). This context is extracted in the same way as the query: as a window, or list of w tokens surrounding the citation left and right. We present our best results, using symmetrical and asymmetrical windows of $w = [(5, 5), (10, 10), (10, 5), (20, 20), (30, 30)]$.
- We build the mixed representations by simply concatenating the internal and external bags-of-words that represent the documents, from which we then build the VSM representation. For this, we combine different window sizes for the *inlink_context* with: *full_text*, *title_abstract* and *passage350*.

5 Results and discussion

Table 1 presents a selection of the most relevant results, where the best result and document representation method of each type is highlighted. We present results for the most relevant parameter values, producing the highest scores of all those tested.

From a close look at internal methods, we can see that the *passage* method with $k = 400$ beats both *full_text* and *title_abstract*, suggesting that a more elaborate way of building a document representation should improve results. This is consistent with previous findings: Gay et al. (2005) had already reported that using selected sections plus captions of figures and title and abstract to build the internal document representation improves the results of their indexing task by 7.4% over just using title and abstract. Similarly, Jimeno-Yepes et al. (2013) showed that automatically generated summaries lead to similar recall and better indexing precision than full-text articles for a keyword-based indexing task.

However, it is immediately clear that purely external methods obtain higher scores than internal ones. The best score of 0.413 is obtained by the *inlink_context* method with a window of 10 tokens left, 5 right, combined with the similarly-sized ex-

³<http://scikit-learn.org>

Method	window5_5	window10_10	window10_5	window20_20	window30_30
<i>Internal methods</i>					
full_text	0.318	0.340	0.337	0.369	0.370
title_abstract	0.296	0.312	0.312	0.322	0.311
passage250	0.343	0.367	0.359	0.388	0.382
passage350	0.346	0.371	0.364	0.388	0.381
passage400	0.348	0.371	0.362	0.391	0.380
<i>External methods</i>					
inlink_context10	0.391	0.406	0.405	0.395	0.387
inlink_context20	0.386	0.406	0.413	0.412	0.402
inlink_context30	0.380	0.403	0.400	0.411	0.404
<i>Mixed methods</i>					
inlink_context_20_full_text	0.367	0.407	0.399	0.431	0.425
inlink_context_20_title_abstract	0.419	0.447	0.441	0.453	0.437
inlink_context_20_passage250	0.420	0.458	0.451	0.469	0.451
inlink_context_10_passage350	0.435	0.465	0.459	0.464	0.450
inlink_context_20_passage350	0.426	0.464	0.456	0.469	0.456

Table 1: Accuracy for each document representation method (rows) and context window size (columns).

traction method for the query (*window10_10*). We find it remarkable that *inlink_context* is superior to internal methods, beating the best (*passage400*) by 0.02 absolute accuracy points. Whether this is because the descriptions of these papers in the contexts of incoming link citations capture the essence or key relevance of the paper, or whether this effect is due to authors reusing their work or to these descriptions originating in a seed paper and being then propagated through the literature, remain interesting research questions that we intend to tackle in future work.

The key finding from our experiments is however that a mixture of internal and external methods beats both individually. The highest score is 0.469, achieved by a combination of *inlink_context_20* and the *passage* method, for a window of $w = 20$, with a tie between using 250 and 350 as values for k (passage size). The small difference in score between parameter values is perhaps not as relevant as the finding that, taken together, mixed methods consistently beat both external and internal methods.

These results also show that the task is far from solved, with the highest accuracy achieved being just under 47%. There is clear room for improvement, which we believe could firstly come from a more targeted extraction of text, both for generating the document representations and for extracting the citation contexts.

Our ultimate goal is matching claims and comparing methods, which would likely benefit from an analysis of the full contents of the document and not just previous citations of it, so in future work we also intend to use the context from the

successful external results as training data for a summarisation stage.

6 Conclusion and future work

In this paper we have presented Citation Resolution: an evaluation method for context-based citation recommendation (CBCR) systems. Our method exploits the implicit human relevance judgements found in existing scientific articles and so does not require purpose-specific human annotation.

We have employed Citation Resolution to test three approaches to building a document representation for a CBCR system: internal (based on the contents of the document), external (based on the surrounding contexts to citations to that document) and mixed (a mixture of the two). Our evaluation shows that: 1) using chunks of a document (passages) as its representation yields better results than using its full text, 2) external methods obtain higher scores than internal ones, and 3) mixed methods yield better results than either in isolation.

We intend to investigate more sophisticated ways of document representation and of extracting a citation’s context. Our ultimate goal is not just to suggest to the author documents that are “relevant” to a specific chunk of the paper (sentence, paragraph, etc.), but to do so with attention to rhetorical structure and thus to citation function. We also aim to apply our evaluation to other document collections in different scientific domains in order to test to what degree these results can be generalized.

References

- Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.
- Kannan Chandrasekaran, Susan Gauch, Praveen Lakkaraju, and Hiep Phuc Luong. 2008. Concept-based document recommendations for citeseer authors. In *Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 83–92. Springer.
- Brian D Davison. 2000. Topical locality in the web. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 272–279. ACM.
- Clifford W Gay, Mehmet Kayaalp, and Alan R Aronson. 2005. Semi-automatic indexing of full text biomedical articles. In *AMIA Annual Symposium Proceedings*, volume 2005, page 271. American Medical Informatics Association.
- Qi He, Jian Pei, Daniel Kifer, Prasenjit Mitra, and Lee Giles. 2010. Context-aware citation recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 421–430. ACM.
- Jing He, Jian-Yun Nie, Yang Lu, and Wayne Xin Zhao. 2012. Position-aligned translation model for citation recommendation. In *String Processing and Information Retrieval*, pages 251–263. Springer.
- Ken Hyland. 2009. *Academic discourse: English in a global context*. Bloomsbury Publishing.
- Antonio J Jimeno-Yepes, Laura Plaza, James G Mork, Alan R Aronson, and Alberto Díaz. 2013. Mesh indexing based on automatically generated summaries. *BMC bioinformatics*, 14(1):208.
- Maria Liakata, Shyamasree Saha, Simon Dobnik, Colin Batchelor, and Dietrich Rebholz-Schuhmann. 2012. Automatic recognition of conceptualization zones in scientific articles and two life science applications. *Bioinformatics*, 28(7):991–1000.
- Michael H MacRoberts and Barbara R MacRoberts. 1996. Problems of citation analysis. *Scientometrics*, 36(3):435–444.
- Anna Ritchie, Simone Teufel, and Stephen Robertson. 2006. Creating a test collection for citation-based ir experiments. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 391–398. Association for Computational Linguistics.
- Anna Ritchie. 2009. Citation context analysis for information retrieval. Technical report, University of Cambridge Computer Laboratory.
- Mark Sanderson. 2010. *Test collection based evaluation of information retrieval systems*. Now Publishers Inc.
- Ulrich Schäfer and Uwe Kasterka. 2010. Scientific authoring support: A tool to navigate in typed citation graphs. In *Proceedings of the NAACL HLT 2010 workshop on computational linguistics and writing: Writing processes and authoring aids*, pages 7–14. Association for Computational Linguistics.
- Simone Teufel, Advaith Siddharthan, and Dan Tidhar. 2006. Automatic classification of citation function. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 103–110. Association for Computational Linguistics.

Hippocratic Abbreviation Expansion

Brian Roark and Richard Sproat

Google, Inc, 79 Ninth Avenue, New York, NY 10011

{roark, rws}@google.com

Abstract

Incorrect normalization of text can be particularly damaging for applications like text-to-speech synthesis (TTS) or typing auto-correction, where the resulting normalization is directly presented to the user, versus feeding downstream applications. In this paper, we focus on abbreviation expansion for TTS, which requires a “do no harm”, high precision approach yielding few expansion errors at the cost of leaving relatively many abbreviations unexpanded. In the context of a large-scale, real-world TTS scenario, we present methods for training classifiers to establish whether a particular expansion is apt. We achieve a large increase in correct abbreviation expansion when combined with the baseline text normalization component of the TTS system, together with a substantial reduction in incorrect expansions.

1 Introduction

Text normalization (Sproat et al., 2001) is an important initial phase for many natural language and speech applications. The basic task of text normalization is to convert *non-standard words* (NSWs) — numbers, abbreviations, dates, etc. — into standard words, though depending on the task and the domain a greater or lesser number of these NSWs may need to be normalized. Perhaps the most demanding such application is text-to-speech synthesis (TTS) since, while for parsing, machine translation and information retrieval it may be acceptable to leave such things as numbers and abbreviations unexpanded, for TTS all tokens need to be *read*, and for that it is necessary to know how to pronounce them. Which normalizations are required depends very much on the application.

What is also very application-dependent is the cost of errors in normalization. For some applications, where the normalized string is an interme-

diated stage in a larger application such as translation or information retrieval, overgeneration of normalized alternatives is often a beneficial strategy, to the extent that it may improve the accuracy of what is eventually being presented to the user. In other applications, such as TTS or typing auto-correction, the resulting normalized string itself is directly presented to the user; hence errors in normalization can have a very high cost relative to leaving tokens unnormalized.

In this paper we concentrate on abbreviations, which we define as alphabetic NSWs that it would be normal to pronounce as their expansion. This class of NSWs is particularly common in personal ads, product reviews, and so forth. For example:

```
home health care svcs stat home health llc  
osceola aquatic ctr stars rating write  
audi vw repair ser quality and customer
```

Each of the examples above contains an abbreviation that, unlike, e.g., conventionalized state abbreviations such as *ca* for *California*, is either only slightly standard (*ctr* for *center*) or not standard at all (*ser* for *service*).

An important principle in text normalization for TTS is *do no harm*. If a system is unable to reliably predict the correct reading for a string, it is better to leave the string alone and have it default to, say, a character-by-character reading, than to expand it to something wrong. This is particularly true in *accessibility* applications for users who rely on TTS for most or all of their information needs. Ideally a navigation system should read *turn on 30N* correctly as *turn on thirty north*; but if it cannot resolve the ambiguity in *30N*, it is far better to read it as *thirty N* than as *thirty Newtons*, since listeners can more easily recover from the first kind of error than the second.

We present methods for learning abbreviation expansion models that favor high precision (incorrect expansions < 2%). Unannotated data is used to collect evidence for contextual disambiguation and to train an abbreviation model. Then a small amount of annotated data is used to build models to determine whether to accept a candidate expansion.

sion of an abbreviation based on these features. The data we report on are taken from Google Maps™ and web pages associated with its map entries, but the methods can be applied to any data source that is relatively abbreviation rich.

We note in passing that similar issues arise in automatic spelling correction work (Wilcox-O’Hearn et al., 2008), where it is better to leave a word alone than to “correct” it wrongly.

2 Related work

There has been a lot of interest in recent years on “normalization” of social media such as Twitter, but that work defines normalization much more broadly than we do here (Xia et al., 2006; Choudhury et al., 2007; Kobus et al., 2008; Beaufort et al., 2010; Kaufmann, 2010; Liu et al., 2011; Pennell and Liu, 2011; Aw and Lee, 2012; Liu et al., 2012a; Liu et al., 2012b; Hassan and Menezes, 2013; Yang and Eisenstein, 2013). There is a good reason for us to focus more narrowly. For Twitter, much of the normalization task involves non-standard language such as *ur website suxx brah* (from Yang and Eisenstein (2013)). Expanding the latter to *your website sucks, brother* certainly normalizes it to standard English, but one could argue that in so doing one is losing information that the writer is trying to convey using an informal style. On the other hand, someone who writes *svc ctr* for *service center* in a product review is probably merely trying to save time and so expanding the abbreviations in that case is neutral with respect to preserving the intent of the original text.

One other difference between the work we report from much of the recent work cited above is that that work focuses on getting high F scores, whereas we are most concerned with getting high precision. While this may seem like a trivial trade off between precision and recall, our goal motivates developing measures that minimize the “risk” of expanding a term, something that is important in an application such as TTS, where one cannot correct a misexpansion after it is spoken.

3 Methods

Since our target application is text-to-speech, we define the task in terms of an existing TTS lexicon. If a word is already in the lexicon, it is left unprocessed, since there is an existing pronunciation for it; if a word is out-of-vocabulary (OOV), we consider expanding it to a word in the lexicon. We consider a possible expansion for an abbreviation to be any word in the lexicon from which the abbreviation can be derived by only deletion of

letters.¹ For present purposes we use the Google English text-to-speech lexicon, consisting of over 430 thousand words. Given an OOV item (possible abbreviation) in context, we make use of features of the context and of the OOV item itself to enumerate and score candidate expansions.

Our data consists of 15.1 billion words of text data from Google Maps™, lower-cased and tokenized to remove punctuation symbols. We used this data in several ways. First, we used it to bootstrap a model for assigning a probability of an abbreviation/expansion pair. Second, we used it to extract contextual n-gram features for predicting possible expansions. Finally, we sampled just over 14 thousand OOV items in context and had them manually labeled with a number of categories, including ‘abbreviation’. OOVs labeled as abbreviations were also labeled with the correct expansion. We present each of these uses in turn.

3.1 Abbreviation modeling

We collect potential abbreviation/full-word pairs by looking for terms that could be abbreviations of full words that occur in the same context. Thus:

the	svc/service	center
heating	clng/cooling	system
dry	clng/cleaning	system

contributes evidence that *svc* is an abbreviation of *service*. Similarly instances of *clng* in contexts that can contain *cooling* or *cleaning* are evidence that *clng* could be an abbreviation of either of these words. (The same contextual information of course is used later on to disambiguate which of the expansions is appropriate for the context.) To compute the initial guess as to what can be a possible abbreviation, a Thrax grammar (Roark et al., 2012) is used that, among other things, specifies that: the abbreviation must start with the same letter as the full word; if a vowel is deleted, all adjacent vowels should also be deleted; consonants may be deleted in a cluster, but not the last one; and a (string) suffix may be deleted.² We count a pair of words as ‘co-occurring’ if they are observed in the same context. For a given context C , e.g., *the__center*, let W_C be the set of words found in that context. Then, for any pair of words u, v , we can assign a pair count based on the count of contexts where both occur:

$$c(u, v) = |\{C : u \in W_C \text{ and } v \in W_C\}|$$

¹We do not deal here with phonetic spellings in abbreviations such as *4get*, or cases where letters have been transposed due to typographical errors (*scv*).

²This Thrax grammar can be found at <http://openfst.cs.nyu.edu/twiki/bin/view/Contrib/ThraxContrib>

blvd boulevard	rd road	yrs years
ca california	fl florida	ctr center
mins minutes	def definitely	ste suite

Table 1: Examples of automatically mined abbreviation/expansion pairs.

Let $c(u)$ be defined as $\sum_v c(u, v)$. From these counts, we can define a 2×2 table and calculate statistics such as the log likelihood statistic (Dunning, 1993), which we use to rank possible abbreviation/expansion pairs. Scores derived from these *type* (rather than *token*) counts highly rank pairs of in-vocabulary words and OOV possible abbreviations that are substitutable in many contexts.

We further filter the potential abbreviations by removing ones that have a lot of potential expansions, where we set the cutoff at 10. This removes mostly short abbreviations that are highly ambiguous. The resulting ranked list of abbreviation expansion pairs is then thresholded before building the abbreviation model (see below) to provide a smaller but more confident training set. For this paper, we used 5-gram contexts (two words on either side) to extract abbreviations and their expansions. See Table 1 for some examples.

Our abbreviation model is a *pair character language model* (LM), also known as a joint multi-gram model (Bisani and Ney, 2008), whereby aligned symbols are treated as a single token and a smoothed n-gram model is estimated. This defines a joint distribution over input and output sequences, and can be efficiently encoded as a weighted finite-state transducer. The extracted abbreviation/expansion pairs are character-aligned and a 7-gram pair character LM is built over the alignments using the OpenGrm n-gram library (Roark et al., 2012). For example:

c:c e:e n:t t:e r:r

Note that, as we’ve defined it, the alignments from abbreviation to expansion allow only identity and insertion, no deletions or substitutions. The cost from this LM, normalized by the length of the expansion, serves as a score for the quality of a putative expansion for an abbreviation.

For a small set of frequent, conventionalized abbreviations (e.g., *ca* for *California* — 63 pairs in total — mainly state abbreviations and similar items), we assign an fixed pair LM score, since these examples are in effect *irregular* cases, where the regularities of the productive abbreviation process do not capture their true cost.

3.2 Contextual features

To predict the expansion given the context, we extract n-gram observations for full words in the TTS lexicon. We do this in two ways. First, we sim-

ply train a smoothed n-gram LM from the data. Because of the size of the data set, this is heavily pruned using relative entropy pruning (Stolcke, 1998). Second, we use log likelihood and log odds ratios (this time using standardly defined n-gram counts) to extract reliable bigram and trigram contexts for words. Space precludes a detailed treatment of these two statistics, but, briefly, both can be derived from contingency table values calculated from the frequencies of (1) the word in the particular context; (2) the word in any context; (3) the context with any word; and (4) all words in the corpus. See Agresti (2002), Dunning (1993) and Monroe et al. (2008) for useful overviews of how to calculate these and other statistics to derive reliable associations. In our case, we use them to derive associations between contexts and words occurring in those contexts. The contexts include trigrams with the target word in any of the three positions, and bigrams with the target word in either position. We filter the set of n-grams based on both their log likelihood and log odds ratios, and provide those scores as features.

3.3 Manual annotations

We randomly selected 14,434 OOVs in their full context, and had them manually annotated as falling within one of 8 categories, along with the expansion if the category was ‘abbreviation’. Note that these are relatively lightweight annotations that do not require extensive linguistics expertise. The abbreviation class is defined as cases where pronouncing as the expansion would be normal. Other categories included letter sequence (expansion would not be normal, e.g., *TV*); partial letter sequence (e.g., *PurePictureTV*); misspelling; leave as is (part of a URL or pronounced as a word, e.g., *NATO*); foreign; don’t know; and junk. Abbreviations accounted for nearly 23% of the cases, and about 3/5 of these abbreviations were instances from the set of 63 conventional abbreviation/expansion pairs mentioned in Section 3.1.

3.4 Abbreviation expansion systems

We have three base systems that we compare here. The first is the hand-built TTS normalization system. This system includes some manually built patterns and an address parser to find common abbreviations that occur in a recognizable context. For example, the grammar covers several hundred city-state combinations, such as *Fairbanks AK*, yielding good performance on such cases.

The other two systems were built using data extracted as described above. Both systems make use of the pair LM outlined in Section 3.1, but differ in how they model context. The first sys-

tem, which we call “N-gram”, uses a pruned Katz (1987) smoothed trigram model. The second system, which we call “SVM”, uses a Support Vector Machine (Cortes and Vapnik, 1995) to classify candidate expansions as being correct or not. For both systems, for any given input OOV, the possible expansion with the highest score is output, along with the decision of whether to expand.

For the “N-gram” system, n-gram negative log probabilities are extracted as follows. Let w_i be the position of the target expansion. We extract the part of the n-gram probability of the string that is not constant across all competing expansions, and normalize by the number of words in that window. Thus the score of the word is:

$$S(w_i) = -\frac{1}{k+1} \sum_{j=i}^{i+k} \log P(w_j | w_{j-1}w_{j-2})$$

In our experiments, $k = 2$ since we have a trigram model, though in cases where the target word is the last word in the string, $k = 1$, because there only the end-of-string symbol must be predicted in addition to the expansion. We then take the Bayesian fusion of this model with the pair LM, by adding them in the log space, to get prediction from both the context and abbreviation model.

For the “SVM” model, we extract features from the log likelihood and log odds scores associated with contextual n-grams, as well as from the pair LM probability and characteristics of the abbreviation itself. We train a linear model on a subset of the annotated data (see section 4). Multiple contextual n-grams may be observed, and we take the maximum log likelihood and log odds scores for each candidate expansion in the observed context. We then quantize these scores down into 16 bins, using the histogram in the training data to define bin thresholds so as to partition the training instances evenly. We also create 16 bins for the pair LM score. A binary feature is defined for each bin that is set to 1 if the current candidate’s score is less than the threshold of that bin, otherwise 0. Thus multiple bin features can be active for a given candidate expansion of the abbreviation.

We also have features that fire for each type of contextual feature (e.g., trigram with expansion as middle word, etc.), including ‘no context’, where none of the trigrams or bigrams from the current example that include the candidate expansion are present in our list. Further, we have features for the length of the abbreviation (shorter abbreviations have more ambiguity, hence are more risky to expand); membership in the list of frequent, conventionalized abbreviations mentioned earlier; and some combinations of these, along with bias

features. We train the model using standard options with Google internal SVM training tools.

Note that the number of n-grams in the two models differs. The N-gram system has around 200M n-grams after pruning; while the SVM model uses around a quarter of that. We also tried a more heavily pruned n-gram model, and the results are only very slightly worse, certainly acceptable for a low-resource scenario.

4 Experimental Results

We split the 3,209 labeled abbreviations into a training set of 2,209 examples and a held aside development set of 1,000 examples. We first evaluate on the development set, then perform a final 10-fold cross validation over the entire set of labeled examples. We evaluate in terms of the percentage of abbreviations that were correctly expanded (true positives, TP) and that were incorrectly expanded (false positives, FP).

Results are shown in Table 2. The first two rows show the baseline TTS system and SVM model. On the development set, both systems have a false positive rate near 3%, i.e., three abbreviations are expanded incorrectly for every 100 examples; and over 50% true positive rate, i.e., more than half of the abbreviations are expanded correctly. To report true and false positive rates for the N-gram system we would need to select an arbitrary decision threshold operating point, unlike the deterministic TTS baseline and the SVM model with its decision threshold of 0. Rather than tune such a meta-parameter to the development set, we instead present an ROC curve comparison of the N-gram and SVM models, and then propose a method for “intersecting” their output without requiring a tuned decision threshold.

Figure 1 presents an ROC curve for the N-gram and SVM systems, and for the simple Bayesian fusion (sum in log space) of their scores. We can see that the SVM model has very high precision for its highest ranked examples, yielding nearly 20% of the correct expansions without any incorrect expansions. However the N-gram system achieves higher true positive rates when the false

System	Percent of abbreviations			
	dev set		full set	
	TP	FP	TP	FP
TTS baseline	55.0	3.1	40.0	3.0
SVM model	52.6	3.3	53.3	2.6
SVM \cap N-gram	50.6	1.1	50.3	0.9
SVM \cap N-gram, then TTS	73.5	1.9	74.5	1.5

Table 2: Results on held-out labeled data, and with final 10-fold cross-validation over the entire labeled set. Percentage of abbreviations expanded correctly (TP) and percentage expanded incorrectly (FP) are reported for each system.

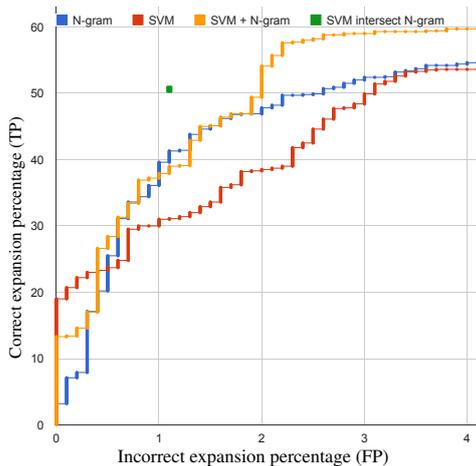


Figure 1: ROC curve plotting true positive (correct expansion) percentages versus false positive (incorrect expansion) percentages for several systems on the development set.

positive rate falls between 1 and 3 percent, though both systems reach roughly the same performance at the SVM’s decision threshold corresponding to around 3.3% false positive rate. The simple combination of their scores achieves strong improvements over either model, with an operating point associated with the SVM decision boundary that yields a couple of points improvement in true positives and a full 1% reduction in false positive rate.

One simple way to combine these two system outputs in a way that does not require tuning a decision threshold is to expand the abbreviation if and only if (1) both the SVM model and the N-gram model agree on the best expansion; and (2) the SVM model score is greater than zero. In a slight abuse of the term ‘intersection’, we call this combination ‘SVM intersect N-gram’ (or ‘SVM \cap N-gram’ in Table 2). Using this approach, our true positive rate on the development set declines a bit to just over 50%, but our false positive rate declines over two full percentage points to 1.1%, yielding a very high precision system.

Taking this very high precision system combination of the N-gram and SVM models, we then combine with the baseline TTS system as follows. First we apply our system, and expand the item if it scores above threshold; for those items left unexpanded, we let the TTS system process it in its own way. In this way, we actually reduce the false positive rate on the development set over the baseline TTS system by over 1% absolute to less than 2%, while also increasing the true positive rate to 73.5%, an increase of 18.5% absolute.

Of course, at test time, we will not know whether an OOV is an abbreviation or not, so we also looked at the performance on the rest of the collected data, to see how often it erroneously suggests an expansion from that set. Of

the 11,157 examples that were hand-labeled as non-abbreviations, our SVM \cap N-gram system expanded 45 items, which is a false positive rate of 0.4% under the assumption that none of them should be expanded. In fact, manual inspection found that 20% of these were correct expansions of abbreviations that had been mis-labeled.

We also experimented with a number of alternative high precision approaches that space precludes our presenting in detail here, including: pruning the number of expansion candidates based on the pair LM score; only allowing abbreviation expansion when at least one extracted n-gram context is present for that expansion in that context; and CART tree (Breiman et al., 1984) training with real valued scores. Some of these yielded very high precision systems, though at the cost of leaving many more abbreviations unexpanded. We found that, for use in combination with the baseline TTS system, large overall reductions in FP rate were achieved by using an initial system with substantially higher TP and somewhat higher FP rates, since far fewer abbreviations were then passed along unexpanded to the baseline system, with its relatively high 3% FP rate.

To ensure that we did not overtune our systems to the development set through experimentation, we performed 10-fold cross validation over the full set of abbreviations. These results are presented in Table 2. Most notably, the TTS baseline system has a much lower true positive rate; yet we find our systems achieve performance very close to that for the development set, so that our final combination with the TTS baseline was actually slightly better than the numbers on the development set.

5 Conclusions

In this paper we have presented methods for high precision abbreviation expansion for a TTS application. The methods are largely self-organizing, using in-domain unannotated data, and depend on only a small amount of annotated data. Since the SVM features relate to general properties of abbreviations, expansions and contexts, the classifier parameters will likely carry over to new (English) domains. We demonstrate that in combination with a hand-built TTS baseline, the methods afford dramatic improvement in the TP rate (to about 74% from a starting point of about 40%) and a reduction of FP to below our goal of 2%.

Acknowledgments

We would like to thank Daan van Esch and the Google Speech Data Operations team for their work on preparing the annotated data. We also thank the reviewers for their comments.

References

- Alan Agresti. 2002. *Categorical data analysis*. John Wiley & Sons, 2nd edition.
- Ai Ti Aw and Lian Hau Lee. 2012. Personalized normalization for a multilingual chat system. In *Proceedings of the ACL 2012 System Demonstrations*, pages 31–36, Jeju Island, Korea, July. Association for Computational Linguistics.
- Richard Beaufort, Sophie Roekhaut, Louise-Amélie Cougnon, and Cédric Fairon. 2010. A hybrid rule/model-based finite-state framework for normalizing SMS messages. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 770–779, Uppsala, Sweden, July. Association for Computational Linguistics.
- Maximilian Bisani and Hermann Ney. 2008. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451.
- Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. 1984. *Classification and Regression Trees*. Wadsworth & Brooks, Pacific Grove CA.
- Monojit Choudhury, Rahul Saraf, Vijit Jain, Sudesha Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *Int. J. Doc. Anal. Recogit.*, 10:157–174.
- Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.
- Ted Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.
- Hany Hassan and Arul Menezes. 2013. Social text normalization using contextual graph random walks. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1577–1586.
- Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recogniser. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401.
- Max Kaufmann. 2010. Syntactic normalization of Twitter messages. In *International Conference on NLP*.
- Catherine Kobus, François Yvon, and Géraldine Damnati. 2008. Normalizing SMS: are two metaphors better than one? In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 441–448, Manchester, UK, August. Coling 2008 Organizing Committee.
- Fei Liu, Fuliang Weng, Bingqing Wang, and Yang Liu. 2011. Insertion, deletion, or substitution? Normalizing text messages without pre-categorization nor supervision. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 71–76, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Fei Liu, Fuliang Weng, and Xiao Jiang. 2012a. A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1035–1044, Jeju Island, Korea, July. Association for Computational Linguistics.
- Xiaohua Liu, Ming Zhou, Xiangyang Zhou, Zhongyang Fu, and Furu Wei. 2012b. Joint inference of named entity recognition and normalization for tweets. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 526–535, Jeju Island, Korea, July. Association for Computational Linguistics.
- Burt L Monroe, Michael P Colaresi, and Kevin M Quinn. 2008. Fightin’ words: Lexical feature selection and evaluation for identifying the content of political conflict. *Political Analysis*, 16(4):372–403.
- Deana Pennell and Yang Liu. 2011. A character-level machine translation approach for normalization of SMS abbreviations. In *IJCNLP*. Papers/pennell-liu3.pdf.
- Brian Roark, Michael Riley, Cyril Allauzen, Terry Tai, and Richard Sproat. 2012. The OpenGrm open-source finite-state grammar software libraries. In *ACL*, Jeju Island, Korea.
- Richard Sproat, Alan Black, Stanley Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. 2001. Normalization of non-standard words. *Computer Speech and Language*, 15(3):287–333.
- Andreas Stolcke. 1998. Entropy-based pruning of backoff language models. In *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, pages 270–274.
- Amber Wilcox-O’Hearn, Graeme Hirst, and Alexander Budanitsky. 2008. Real-word spelling correction with trigrams: A reconsideration of the Mays, Damerau, and Mercer model. In *CICLing 2008*, volume 4919 of *LNCS*, pages 605–616, Berlin. Springer.
- Yunqing Xia, Kam-Fai Wong, and Wenjie Li. 2006. A phonetic-based approach to Chinese chat text normalization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 993–1000, Sydney, Australia, July. Association for Computational Linguistics.
- Yi Yang and Jacob Eisenstein. 2013. A log-linear model for unsupervised text normalization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 61–72.

Unsupervised Feature Learning for Visual Sign Language Identification

Binyam Gebrekidan Gebre¹, Onno Crasborn², Peter Wittenburg¹,
Sebastian Drude¹, Tom Heskes²

¹Max Planck Institute for Psycholinguistics, ²Radboud University Nijmegen
bingeb@mpi.nl, o.crasborn@let.ru.nl, peter.wittenburg@mpi.nl,
sebastian.drude@mpi.nl, t.heskes@science.ru.nl

Abstract

Prior research on language identification focused primarily on text and speech. In this paper, we focus on the visual modality and present a method for identifying sign languages solely from short video samples. The method is trained on unlabelled video data (unsupervised feature learning) and using these features, it is trained to discriminate between six sign languages (supervised learning). We ran experiments on short video samples involving 30 signers (about 6 hours in total). Using leave-one-signer-out cross-validation, our evaluation shows an average best accuracy of 84%. Given that sign languages are under-resourced, unsupervised feature learning techniques are the right tools and our results indicate that this is realistic for sign language identification.

1 Introduction

The task of automatic language identification is to quickly identify the identity of the language given utterances. Performing this task is key in applications involving multiple languages such as machine translation and information retrieval (e.g. metadata creation for large audiovisual archives).

Prior research on language identification is heavily biased towards written and spoken languages (Dunning, 1994; Zissman, 1996; Li et al., 2007; Singer et al., 2012). While language identification in signed languages is yet to be studied, significant progress has been recorded for written and spoken languages.

Written languages can be identified to about 99% accuracy using Markov models (Dunning, 1994). This accuracy is so high that current research has shifted to related more challenging problems: language variety identification (Zampieri and Gebre, 2012), native language identification (Tetreault et al., 2013) and identification at the extremes of scales; many more languages,

smaller training data, shorter document lengths (Baldwin and Lui, 2010).

Spoken languages can be identified to accuracies that range from 79-98% using different models (Zissman, 1996; Singer et al., 2003). The methods used in spoken language identification have also been extended to a related class of problems: native accent identification (Chen et al., 2001; Choueiter et al., 2008; Wu et al., 2010) and foreign accent identification (Teixeira et al., 1996).

While some work exists on sign language recognition¹ (Starner and Pentland, 1997; Starner et al., 1998; Gavrilu, 1999; Cooper et al., 2012), very little research exists on sign language identification except for the work by (Gebre et al., 2013), where it is shown that sign language identification can be done using linguistically motivated features. Accuracies of 78% and 95% are reported on signer independent and signer dependent identification of two sign languages.

This paper has two goals. First, to present a method to identify sign languages using features learned by unsupervised techniques (Hinton and Salakhutdinov, 2006; Coates et al., 2011). Second, to evaluate the method on six sign languages under different conditions.

Our contributions: *a*) show that unsupervised feature learning techniques, currently popular in many pattern recognition problems, also work for visual sign languages. More specifically, we show how K-means and sparse autoencoder can be used to learn features for sign language identification. *b*) demonstrate the impact on performance of varying the number of features (aka, feature maps or filter sizes), the patch dimensions (from 2D to 3D) and the number of frames (video length).

¹There is a difference between sign language recognition and identification. Sign language recognition is the recognition of the meaning of the signs in a given known sign language, whereas sign language identification is the recognition of the sign language itself from given signs.

2 The challenges in sign language identification

The challenges in sign language identification arise from three sources as described below.

2.1 Iconicity in sign languages

The relationship between forms and meanings are not totally arbitrary (Perniss et al., 2010). Both signed and spoken languages manifest iconicity, that is forms of words or signs are somehow motivated by the meaning of the word or sign. While sign languages show a lot of iconicity in the lexicon (Taub, 2001), this has not led to a universal sign language. The same concept can be iconically realised by the manual articulators in a way that conforms to the phonological regularities of the languages, but still lead to different sign forms.

Iconicity is also used in the morphosyntax and discourse structure of all sign languages, however, and there we see many similarities between sign languages. Both real-world and imaginary objects and locations are visualised in the space in front of the signer, and can have an impact on the articulation of signs in various ways. Also, the use of constructed action appears to be used in many sign languages in similar ways. The same holds for the rich use of non-manual articulators in sentences and the limited role of facial expressions in the lexicon: these too make sign languages across the world very similar in appearance, even though the meaning of specific articulations may differ (Crasborn, 2006).

2.2 Differences between signers

Just as speakers have different voices unique to each individual, signers have also different signing styles that are likely unique to each individual. Signers' uniqueness results from how they articulate the shapes and movements that are specified by the linguistic structure of the language. The variability between signers either in terms of physical properties (hand sizes, colors, etc) or in terms of articulation (movements) is such that it does not affect the understanding of the sign language by humans, but that it may be difficult for machines to generalize over multiple individuals. At present we do not know whether the differences between signers using the same language are of a similar or different nature than the differences between different languages. At the level of phonology, there are few differences between sign languages, but

the differences in the phonetic realization of words (their articulation) may be much larger.

2.3 Diverse environments

The visual 'activity' of signing comes in a context of a specific environment. This environment can include the visual background and camera noises. The background objects of the video may also include dynamic objects – increasing the ambiguity of signing activity. The properties and configurations of the camera induce variations of scale, translation, rotation, view, occlusion, etc. These variations coupled with lighting conditions may introduce noise. These challenges are by no means specific to sign interaction, and are found in many other computer vision tasks.

3 Method

Our method performs two important tasks. First, it learns a feature representation from patches of unlabelled raw video data (Hinton and Salakhutdinov, 2006; Coates et al., 2011). Second, it looks for activations of the learned representation (by convolution) and uses these activations to learn a classifier to discriminate between sign languages.

3.1 Unsupervised feature learning

Given samples of sign language videos (unknown sign language with one signer per video), our system performs the following steps to learn a feature representation (note that these video samples are separate from the video samples that are later used for classifier learning or testing):

1. **Extract patches.** Extract small videos (hereafter called patches) randomly from anywhere in the video samples. We fix the size of the patches such that they all have r rows, c columns and f frames and we extract patches m times. This gives us $\mathbf{X} = \{x^{(1)}, x^{(1)}, \dots, x^{(m)}\}$, where $x^{(i)} \in R^N$ and $N = r * c * f$ (the size of a patch). For our experiments, we extract 100,000 patches of size $15 * 15 * 1$ (2D) and $15 * 15 * 2$ (3D).
2. **Normalize the patches.** There is evidence that normalization and whitening (Hyvärinen and Oja, 2000) improve performance in unsupervised feature learning (Coates et al., 2011). We therefore normalize every patch $x^{(i)}$ by subtracting the mean and dividing by

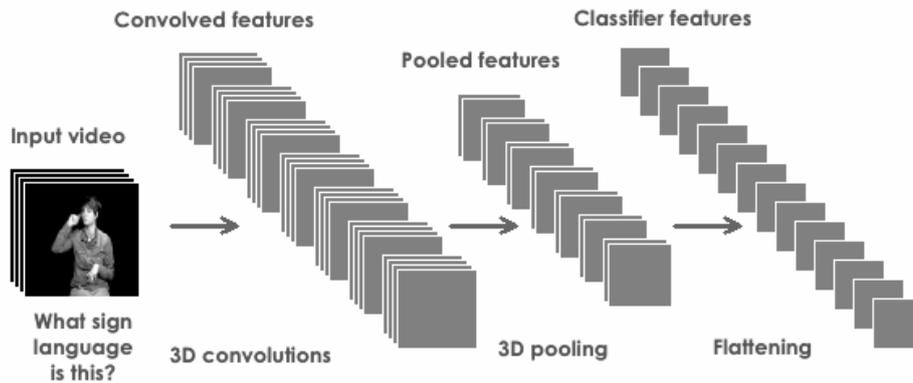


Figure 1: Illustration of feature extraction: convolution and pooling.

the standard deviation of its elements. For visual data, normalization corresponds to local brightness and contrast normalization.

3. **Learn a feature-mapping.** Our unsupervised algorithm takes in the normalized and whitened dataset $\mathbf{X} = \{x^{(1)}, x^{(1)}, \dots, x^{(m)}\}$ and maps each input vector $x^{(i)}$ to a new feature vector of K features ($f : R^N \rightarrow R^K$). We use two unsupervised learning algorithms
- a) K-means
 - b) sparse autoencoders.

- (a) **K-means clustering:** we train K-means to learn K $c^{(k)}$ centroids that minimize the distance between data points and their nearest centroids (Coates and Ng, 2012). Given the learned centroids $c^{(k)}$, we measure the distance of each data point (patch) to the centroids. Naturally, the data points are at different distances to each centroid, we keep the distances that are below the average of the distances and we set the other to zero:

$$f_k(x) = \max\{0, \mu(z) - z_k\} \quad (1)$$

where $z_k = \|x - c^{(k)}\|^2$ and $\mu(z)$ is the mean of the elements of z .

- (b) **Sparse autoencoder:** we train a single layer autoencoder with K hidden nodes using backpropagation to minimize squared reconstruction error. At the hidden layer, the features are mapped using a rectified linear (ReLU) function (Maas et al., 2013) as follows:

$$f(x) = g(Wx + b) \quad (2)$$

where $g(z) = \max(z, 0)$. Note that ReL nodes have advantages over sigmoid or tanh functions; they create sparse representations and are suitable for naturally sparse data (Glorot et al., 2011).

From K-means, we get K R^N centroids and from the sparse autoencoder, we get $W \in R^{K \times N}$ and $b \in R^K$ filters. We call both the centroids and filters as the learned features.

3.2 Classifier learning

Given the learned features, the feature mapping functions and a set of labeled training videos, we extract features as follows:

1. **Convolutional extraction:** Extract features from equally spaced sub-patches covering the video sample.
2. **Pooling:** Pool features together over four non-overlapping regions of the input video to reduce the number of features. We perform max pooling for K-means and mean pooling for the sparse autoencoder over 2D regions (per frame) and over 3D regions (per all sequence of frames).
3. **Learning:** Learn a linear classifier to predict the labels given the feature vectors. We use logistic regression classifier and support vector machines (Pedregosa et al., 2011).

The extraction of classifier features through convolution and pooling is illustrated in figure 1.

4 Experiments

4.1 Datasets

Our experimental data consist of videos of 30 signers equally divided between six sign languages: British sign language (BSL), Danish (DSL), French Belgian (FBSL), Flemish (FSL), Greek (GSL), and Dutch (NGT). The data for the unsupervised feature learning comes from half of the BSL and GSL videos in the Dicta-Sign corpus². Part of the other half, involving 5 signers, is used along with the other sign language videos for learning and testing classifiers.

For the unsupervised feature learning, two types of patches are created: 2D dimensions ($15 * 15$) and 3D ($15 * 15 * 2$). Each type consists of randomly selected 100,000 patches and involves 16 different signers. For the supervised learning, 200 videos (consisting of 1 through 4 frames taken at a step of 2) are randomly sampled per sign language per signer (for a total of 6,000 samples).

4.2 Data preprocessing

The data preprocessing stage has two goals.

First, to remove any non-signing signals that remain constant within videos of a single sign language but that are different across sign languages. For example, if the background of the videos is different across sign languages, then classifying the sign languages could be done with perfection by using signals from the background. To avoid this problem, we removed the background by using background subtraction techniques and manually selected thresholds.

The second reason for data preprocessing is to make the input size smaller and uniform. The videos are colored and their resolutions vary from $320 * 180$ to $720 * 576$. We converted the videos to grayscale and resized their heights to 144 and cropped out the central $144 * 144$ patches.

4.3 Evaluation

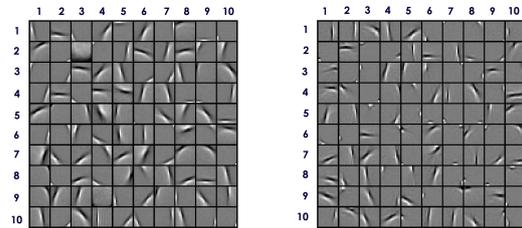
We evaluate our system in terms of average accuracies. We train and test our system in leave-one-signer-out cross-validation, where videos from four signers are used for training and videos of the remaining signer are used for testing. Classification algorithms are used with their default settings and the classification strategy is **one-vs.-rest**.

²<http://www.dictasign.eu/>

5 Results and Discussion

Our best average accuracy (84.03%) is obtained using 500 K-means features which are extracted over four frames (taken at a step of 2). This accuracy obtained for six languages is much higher than the 78% accuracy obtained for two sign languages (Gebre et al., 2013). The latter uses linguistically motivated features that are extracted over video lengths of at least 10 seconds. Our system uses learned features that are extracted over much smaller video lengths (about half a second).

All classification accuracies are presented in table 5 for 2D and table 5 for 3D. Classification confusions are shown in table 5. Figure 2 shows features learned by K-means and sparse autoencoder.



(a) K-means features (b) SAE features

Figure 2: All 100 features learned from 100,000 patches of size $15 * 15$. K-means learned relatively more curving edges than the sparse auto encoder.

K	K-means			Sparse Autoencoder		
	LR-L1	LR-L2	SVM	LR-L1	LR-L2	SVM
# of frames = 1						
100	69.23	70.60	67.42	73.85	74.53	71.8
300	76.08	77.37	74.80	72.27	70.67	68.90
500	83.03	79.88	77.92	67.50	69.38	66.20
# of frames = 2						
100	71.15	72.07	67.42	72.78	74.62	72.08
300	77.33	78.27	76.60	71.85	71.07	68.27
500	83.58	79.50	79.90	67.73	70.15	66.45
# of frames = 3						
100	71.42	73.10	67.82	65.70	67.52	63.68
300	78.40	78.57	76.50	72.53	71.68	68.18
500	83.48	80.05	80.57	67.85	70.85	66.77
# of frames = 4						
100	71.88	73.05	68.70	64.93	67.48	63.80
300	79.32	78.65	76.42	72.27	72.18	68.35
500	84.03	80.38	80.50	68.25	71.57	67.27

K = # of features, SVM = SVM with linear kernel
LR-L? = Logistic Regression with L1 and L2 penalty

Table 1: 2D filters ($15 * 15$): Leave-one-signer-out cross-validation average accuracies.

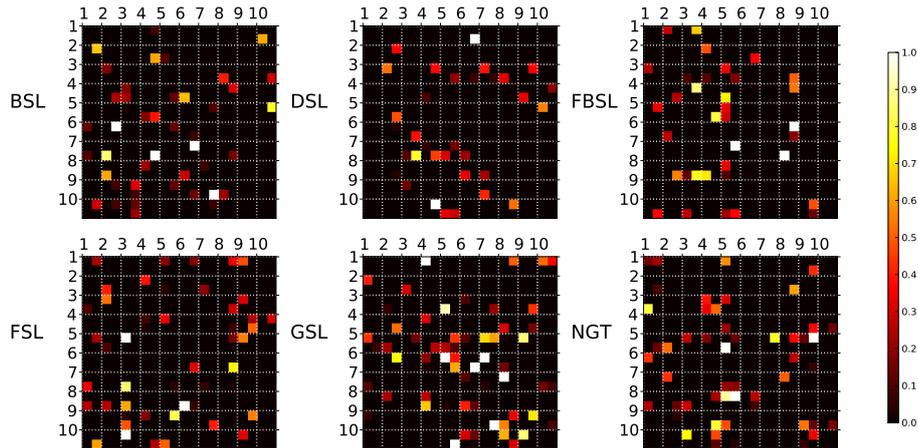


Figure 3: Visualization of coefficients of Lasso (logistic regression with L1 penalty) for each sign language with respect to each of the 100 filters of the sparse autoencoder. The 100 filters are shown in figure 2(b). Each grid cell represents a frame and each filter is activated in 4 non-overlapping pooling regions.

K	K-means			Sparse Autoencoder		
	LR-L1	LR-L2	SVM	LR-L1	LR-L2	SVM
# of frames = 2						
100	70.63	69.62	68.87	67.40	66.53	65.73
300	73.73	74.05	73.03	72.83	73.48	70.52
500	75.30	76.53	75.40	72.28	74.65	68.72
# of frames = 3						
100	72.48	73.30	70.33	68.68	67.40	68.33
300	74.78	74.95	74.77	74.20	74.72	70.85
500	77.27	77.50	76.17	72.40	75.45	69.42
# of frames = 4						
100	74.85	73.97	69.23	68.68	67.80	68.80
300	76.23	76.58	74.08	74.43	75.20	70.65
500	79.08	78.63	76.63	73.50	76.23	70.53

Table 2: 3D filters ($15 * 15 * 2$): Leave-one-signer-out cross-validation average accuracies.

	BSL	DSL	FBSL	FSL	GSL	NGT
BSL	56.11	2.98	1.79	3.38	24.11	11.63
DSL	2.87	92.37	0.95	0.46	3.16	0.18
FBSL	1.48	1.96	79.04	4.69	6.62	6.21
FSL	6.96	2.96	2.06	60.81	18.15	9.07
GSL	5.50	2.55	1.67	2.57	86.05	1.65
NGT	9.08	1.33	3.98	18.76	4.41	62.44

Table 3: Confusion matrix – confusions averaged over all settings for K-means and sparse autoencoder with 2D and 3D filters (i.e. for all # of frames, all filter sizes and all classifiers).

Tables 5 and 5 indicate that K-means performs better with 2D filters and that sparse autoencoder performs better with 3D filters. Note that features from 2D filters are pooled over each frame and

concatenated whereas, features from 3D filters are pooled over all frames.

Which filters are active for which language? Figure 3 shows visualization of the strength of filter activation for each sign language. The figure shows what Lasso looks for when it identifies any of the six sign languages.

6 Conclusions and Future Work

Given that sign languages are under-resourced, unsupervised feature learning techniques are the right tools and our results show that this is realistic for sign language identification.

Future work can extend this work in two directions: 1) by increasing the number of sign languages and signers to check the stability of the learned feature activations and to relate these to iconicity and signer differences 2) by comparing our method with deep learning techniques. In our experiments, we used a single hidden layer of features, but it is worth researching into deeper layers to improve performance and gain more insight into the hierarchical composition of features.

Other questions for future work. How good are human beings at identifying sign languages? Can a machine be used to evaluate the quality of sign language interpreters by comparing them to a native language model? The latter question is particularly important given what happened at the Nelson Mandela’s memorial service³.

³<http://www.youtube.com/watch?v=X-DxGoIVUWo>

References

- Timothy Baldwin and Marco Lui. 2010. Language identification: The long and the short of the matter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 229–237. Association for Computational Linguistics.
- Tao Chen, Chao Huang, E. Chang, and Jingchun Wang. 2001. Automatic accent identification using gaussian mixture models. In *Automatic Speech Recognition and Understanding, 2001. ASRU '01. IEEE Workshop on*, pages 343–346.
- Ghinwa Choueiter, Geoffrey Zweig, and Patrick Nguyen. 2008. An empirical study of automatic accent classification. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 4265–4268. IEEE.
- Adam Coates and Andrew Y Ng. 2012. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade*, pages 561–580. Springer.
- Adam Coates, Andrew Y Ng, and Honglak Lee. 2011. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics*, pages 215–223.
- H. Cooper, E.J. Ong, N. Pugeault, and R. Bowden. 2012. Sign language recognition using sub-units. *Journal of Machine Learning Research*, 13:2205–2231.
- Onno Crasborn, 2006. *Nonmanual structures in sign languages*, volume 8, pages 668–672. Elsevier, Oxford.
- T. Dunning. 1994. *Statistical identification of language*. Computing Research Laboratory, New Mexico State University.
- Dariu M Gavrilă. 1999. The visual analysis of human movement: A survey. *Computer vision and image understanding*, 73(1):82–98.
- Binyam Gebrekidan Gebre, Peter Wittenburg, and Tom Heskes. 2013. Automatic sign language identification. In *Proceedings of ICIP 2013*.
- Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP Volume*, volume 15, pages 315–323.
- Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.
- Aapo Hyvärinen and Erkki Oja. 2000. Independent component analysis: algorithms and applications. *Neural networks*, 13(4):411–430.
- Haizhou Li, Bin Ma, and Chin-Hui Lee. 2007. A vector space modeling approach to spoken language identification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(1):271–284.
- Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the ICML*.
- Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830.
- Pamela Perniss, Robin L Thompson, and Gabriella Vigliocco. 2010. Iconicity as a general property of language: evidence from spoken and signed languages. *Frontiers in psychology*, 1.
- E. Singer, PA Torres-Carrasquillo, TP Gleason, WM Campbell, and D.A. Reynolds. 2003. Acoustic, phonetic, and discriminative approaches to automatic language identification. In *Proc. Eurospeech*, volume 9.
- E. Singer, P. Torres-Carrasquillo, D. Reynolds, A. McCree, F. Richardson, N. Dehak, and D. Sturim. 2012. The mitll nist lre 2011 language recognition system. In *Odyssey 2012-The Speaker and Language Recognition Workshop*.
- Thad Starner and Alex Pentland. 1997. Real-time american sign language recognition from video using hidden markov models. In *Motion-Based Recognition*, pages 227–243. Springer.
- Thad Starner, Joshua Weaver, and Alex Pentland. 1998. Real-time american sign language recognition using desk and wearable computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1371–1375.
- Sarah Taub. 2001. *Language from the body: iconicity and metaphor in American Sign Language*. Cambridge University Press, Cambridge.
- C. Teixeira, I. Trancoso, and A. Serralheiro. 1996. Accent identification. In *Spoken Language, 1996. IC-SLP 96. Proceedings., Fourth International Conference on*, volume 3, pages 1784–1787 vol.3.
- Joel Tetreault, Daniel Blanchard, and Aoife Cahill. 2013. A report on the first native language identification shared task. *NAACL/HLT 2013*, page 48.
- Tingyao Wu, Jacques Duchateau, Jean-Pierre Martens, and Dirk Van Compernelle. 2010. Feature subset selection for improved native accent identification. *Speech Communication*, 52(2):83–98.
- Marcos Zampieri and Binyam Gebrekidan Gebre. 2012. Automatic identification of language varieties: The case of portuguese. In *Proceedings of KONVENS*, pages 233–237.

M.A. Zissman. 1996. Comparison of four approaches to automatic language identification of telephone speech. *IEEE Transactions on Speech and Audio Processing*, 4(1):31–44.

Experiments with crowdsourced re-annotation of a POS tagging data set

Dirk Hovy, Barbara Plank, and Anders Søgaard

Center for Language Technology

University of Copenhagen

Njalsgade 140, 2300 Copenhagen

{dirk|bplank}@cst.dk, soegaard@hum.ku.dk

Abstract

Crowdsourcing lets us collect multiple annotations for an item from several annotators. Typically, these are annotations for non-sequential classification tasks. While there has been some work on crowdsourcing named entity annotations, researchers have largely assumed that syntactic tasks such as part-of-speech (POS) tagging cannot be crowdsourced. This paper shows that workers *can* actually annotate sequential data almost as well as experts. Further, we show that the models learned from crowdsourced annotations fare as well as the models learned from expert annotations in downstream tasks.

1 Introduction

Training good predictive NLP models typically requires annotated data, but getting professional annotators to build useful data sets is often time-consuming and expensive. Snow et al. (2008) showed, however, that crowdsourced annotations can produce similar results to annotations made by experts. Crowdsourcing services such as Amazon’s Mechanical Turk has since been successfully used for various annotation tasks in NLP (Jha et al., 2010; Callison-Burch and Dredze, 2010).

However, most applications of crowdsourcing in NLP have been concerned with classification problems, such as document classification and constructing lexica (Callison-Burch and Dredze, 2010). A large part of NLP problems, however, are structured prediction tasks. Typically, sequence labeling tasks employ a larger set of labels than classification problems, as well as complex interactions between the annotations. Disagreement among annotators is therefore potentially higher, and the task of annotating structured data thus harder.

Only a few recent studies have investigated crowdsourcing sequential tasks; specifically, named entity recognition (Finin et al., 2010; Rodrigues et al., 2013). Results for this are good. However, named entities typically use only few labels (LOC, ORG, and PER), and the data contains mostly non-entities, so the complexity is manageable. The question of whether a more linguistically involved structured task like part-of-speech (POS) tagging can be crowdsourced has remained largely unaddressed.¹

In this paper, we investigate how well lay annotators can produce POS labels for Twitter data. In our setup, we present annotators with one word at a time, with a minimal surrounding context (two words to each side). Our choice of annotating Twitter data is not coincidental: with the short-lived nature of Twitter messages, models quickly lose predictive power (Eisenstein, 2013), and re-training models on new samples of more representative data becomes necessary. Expensive professional annotation may be prohibitive for keeping NLP models up-to-date with linguistic and topical changes on Twitter. We use a minimum of instructions and require few qualifications.

Obviously, lay annotation is generally less reliable than professional annotation. It is therefore common to aggregate over multiple annotations for the same item to get more robust annotations. In this paper we compare two aggregation schemes, namely majority voting (MV) and MACE (Hovy et al., 2013). We also show how we can use Wiktionary, a crowdsourced lexicon, to filter crowdsourced annotations. We evaluate the annotations in several ways: (a) by testing their accuracy with respect to a gold standard, (b) by evaluating the performance of POS models trained on

¹One of the reviewers alerted us to an unpublished masters thesis, which uses pre-annotation to reduce tagging to fewer multiple-choice questions. See Related Work section for details.

the annotations across several existing data sets, as well as (c) by applying our models in downstream tasks. We show that with minimal context and annotation effort, we can produce structured annotations of near-expert quality. We also show that these annotations lead to better POS tagging models than previous models learned from crowd-sourced lexicons (Li et al., 2012). Finally, we show that models learned from these annotations are competitive with models learned from expert annotations on various downstream tasks.

2 Our Approach

We crowdsource the training section of the data from Gimpel et al. (2011)² with POS tags. We use Crowdfunder,³ to collect five annotations for each word, and then find the most likely label for each word among the possible annotations. See Figure 1 for an example. If the correct label is not among the annotations, we are unable to recover the correct answer. This was the case for 1497 instances in our data (cf. the token “:” in the example). We thus report on oracle score, i.e., the best label sequence that could possibly be found, which is correct except for the missing tokens. Note that while we report agreement between the crowd-sourced annotations and the crowdsourced annotations, our main evaluations are based on models learned from expert vs. crowdsourced annotations and downstream applications thereof (chunking and NER). We take care in evaluating our models across different data sets to avoid biasing our evaluations to particular annotations. All the data sets used in our experiments are publicly available at <http://lowlands.ku.dk/results/>.

x	Z	y
@USER	NOUN, NOUN, X, NOUN, -, NOUN	NOUN
:	., ., -, ., ., ., .	X
I	PRON, NOUN, PRON, NOUN, PRON, -	PRON
owe	VERB, VERB, -, VERB, VERB, VERB	VERB
U	PRON, X, -, NOUN, NOUN, PRON	PRON

$\theta = 0.9, 0.4, 0.2, 0.8, 0.8, 0.9$

Figure 1: Five annotations per token, supplied by 6 different annotators (– = missing annotation), gold label y . θ = competence values for each annotator.

²<http://www.ark.cs.cmu.edu/TweetNLP/>

³<http://crowdfunder.com>

3 Crowdsourcing Sequential Annotation

In order to use the annotations to train models that can be applied across various data sets, i.e., making out-of-sample evaluation possible (see Section 5), we follow Hovy et al. (2014) in using the universal tag set (Petrov et al., 2012) with 12 labels.



Figure 2: Screen shot of the annotation interface on Crowdfunder

Annotators were given a bold-faced word with two words on either side and asked to select the most appropriate tag from a drop down menu. For each tag, we spell out the name of the syntactic category, and provide a few example words. See Figure 2 for a screenshot of the interface. Annotators were also told that words can belong to several classes, depending on the context. No additional guidelines were given.

Only trusted annotators (in Crowdfunder: Bronze skills) that had answered correctly on 4 gold tokens (randomly chosen from a set of 20 gold tokens provided by the authors) were allowed to submit annotations. In total, 177 individual annotators supplied answers. We paid annotators a reward of \$0.05 for 10 tokens. The full data set contains 14,619 tokens. Completion of the task took slightly less than 10 days. Contributors were very satisfied with the task (4.5 on a scale from 1 to 5). In particular, they felt instructions were clear (4.4/5), and that the pay was reasonable (4.1/5).

4 Label Aggregation

After collecting the annotations, we need to aggregate the annotations to derive a single answer for each token. In the simplest scheme, we choose the majority label, i.e., the label picked by most annotators. In case of ties, we select the final label at random. Since this is a stochastic process, we average results over 100 runs. We refer to this as MAJORITY VOTING (MV). Note that in MV we trust all annotators to the same degree. However, crowdsourcing attracts people with different mo-

tives, and not all of them are equally reliable—even the ones with Bronze level. Ideally, we would like to factor this into our decision process.

We use MACE⁴ (Hovy et al., 2013) as our second scheme to learn both the most likely answer and a competence estimate for each of the annotators. MACE treats annotator competence and the correct answer as hidden variables and estimates their parameters via EM (Dempster et al., 1977). We use MACE with default parameter settings to give us the weighted average for each annotated example.

Finally, we also tried applying the joint learning scheme in Rodrigues et al. (2013), but their scheme requires that entire sequences are annotated by the same annotators, which we don't have, and it expects BIO sequences, rather than POS tags.

Dictionaries Decoding tasks profit from the use of dictionaries (Merialdo, 1994; Johnson, 2007; Ravi and Knight, 2009) by restricting the number of tags that need to be considered for each word, also known as *type constraints* (Täckström et al., 2013). We follow Li et al. (2012) in including Wiktionary information as type constraints into our decoding: if a word is found in Wiktionary, we disregard all annotations that are not licensed by the dictionary entry. If the word is not found in Wiktionary, or if none of its annotations is licensed by Wiktionary, we keep the original annotations. Since we aggregate annotations independently (unlike Viterbi decoding), we basically use Wiktionary as a pre-filtering step, such that MV and MACE only operate on the reduced annotations.

5 Experiments

Each of the two aggregation schemes above produces a final label sequence \hat{y} for our training corpus. We evaluate the resulting annotated data in three ways.

1. We compare \hat{y} to the available expert annotation on the *training* data. This tells us how similar lay annotation is to professional annotation.

2. Ultimately, we want to use structured annotations for supervised training, where annotation quality influences model performance on held-out *test* data. To test this, we train a CRF model (Lafferty et al., 2001) with simple orthographic features and word clusters (Owoputi et al., 2013)

⁴<http://www.isi.edu/publications/licensed-sw/mace/>

on the annotated Twitter data described in Gimpel et al. (2011). Leaving out the dedicated test set to avoid in-sample bias, we evaluate our models across three data sets: RITTER (the 10% test split of the data in Ritter et al. (2011) used in Derczynski et al. (2013)), the test set from Foster et al. (2011), and the data set described in Hovy et al. (2014).

We will make the preprocessed data sets available to the public to facilitate comparison. In addition to a supervised model trained on expert annotations, we compare our tagging accuracy with that of a weakly supervised system (Li et al., 2012) re-trained on 400,000 unlabeled tweets to adapt to Twitter, but using a crowdsourced lexicon, namely Wiktionary, to constrain inference. We use parameter settings from Li et al. (2012), as well as their Wikipedia dump, available from their project website.⁵

3. POS tagging is often the first step for further analysis, such as chunking, parsing, etc. We test the downstream performance of the POS models from the previous step on chunking and NER. We use the models to annotate the training data portion of each task with POS tags, and use them as features in a chunking and NER model. For both tasks, we train a CRF model on the respective (POS-augmented) training set, and evaluate it on several held-out test sets. For chunking, we use the test sets from Foster et al. (2011) and Ritter et al. (2011) (with the splits from Derczynski et al. (2013)). For NER, we use data from Finin et al. (2010) and again Ritter et al. (2011). For chunking, we follow Sha and Pereira (2003) for the set of features, including token and POS information. For NER, we use standard features, including POS tags (from the previous experiments), indicators for hyphens, digits, single quotes, upper/lowercase, 3-character prefix and suffix information, and Brown word cluster features⁶ with 2,4,8,16 bitstring prefixes estimated from a large Twitter corpus (Owoputi et al., 2013). We report macro-averages over all these data sets.

6 Results

Agreement with expert annotators Table 1 shows the accuracy of each aggregation compared to the gold labels. The crowdsourced annotations

⁵<https://code.google.com/p/wikily-supervised-pos-tagger/>

⁶<http://www.ark.cs.cmu.edu/TweetNLP/>

majority	79.54
MACE-EM	79.89
majority+Wiktionary	80.58
MACE-EM+Wiktionary	80.75
oracle	89.63

Table 1: Accuracy (%) of different annotations wrt gold data

aggregated using MV agree with the expert annotations in 79.54% of the cases. If we pre-filter the data using Wiktionary, the agreement becomes 80.58%. MACE leads to higher agreement with expert annotations under both conditions (79.89 and 80.75). The small difference indicates that annotators are consistent and largely reliable, thus confirming the Bronze-level qualification we required. Both schemes cannot recover the correct answer for the 1497 cases where none of the crowdsourced labels matched the gold label, i.e. $y \notin \mathbf{Z}_i$. The best possible result either of them could achieve (the *oracle*) would be matching all but the missing labels, an agreement of 89.63%.

Most of the cases where the correct label was not among the annotations belong to a small set of confusions. The most frequent was mislabeling “:” and “...”, both mapped to *X*. Annotators mostly decided to label these tokens as punctuation (.). They also predominantly labeled *your*, *my* and *this* as *PRON* (for the former two), and a variety of labels for the latter, when the gold label is *DET*.

	RITTER	FOSTER	HOVY
Li et al. (2012)	73.8	77.4	79.7
MV	80.5	81.6	83.7
MACE	80.4	81.7	82.6
MV+Wik	80.4	82.1	83.7
MACE+Wik	80.5	81.9	83.7
Upper bounds			
oracle	82.4	83.7	85.1
gold	82.6	84.7	86.8

Table 2: POS tagging accuracies (%).

Effect on POS Tagging Accuracy Usually, we don’t want to match a gold standard, but we rather want to create new annotated training data. Crowdsourcing matches our gold standard to about 80%, but the question remains how useful this data is when training models on it. After all, inter-annotator agreement among professional an-

notators on this task is only around 90% (Gimpel et al., 2011; Hovy et al., 2014). In order to evaluate how much each aggregation scheme influences tagging performance of the resulting model, we train separate models on each scheme’s annotations and test on the same four data sets. Table 2 shows the results. Note that the differences between the four schemes are insignificant. More importantly, however, POS tagging accuracy using crowdsourced annotations are on average *only 2.6% worse* than gold using professional annotations. On the other hand, performance is *much better* than the weakly supervised approach by Li et al. (2012), which only relies on a crowdsourced POS lexicon.

POS model from	CHUNKING	NER
MV	74.80	75.74
MACE	75.04	75.83
MV+Wik	75.86	76.08
MACE+Wik	75.86	76.15
Upper bounds		
oracle	76.22	75.85
gold	79.97	75.81

Table 3: Downstream accuracy for chunking (l) and NER (r) of models using POS.

Downstream Performance Table 3 shows the accuracy when using the POS models trained in the previous evaluation step. Note that we present the average over the two data sets used for each task. Note also how the Wiktionary constraints lead to improvements in downstream performance. In chunking, we see that using the crowdsourced annotations leads to worse performance than using the professional annotations. For NER, however, we find that some of the POS taggers trained on aggregated data produce better NER performance than POS taggers trained on expert-annotated gold data. Since the only difference between models are the respective POS features, the results suggest that at least for some tasks, POS taggers learned from crowdsourced annotations may be *as good* as those learned from expert annotations.

7 Related Work

There is considerable work in the literature on modeling answer correctness and annotator competence as latent variables (Dawid and Skene,

1979; Smyth et al., 1995; Carpenter, 2008; Whitehill et al., 2009; Welinder et al., 2010; Yan et al., 2010; Raykar and Yu, 2012). Rodrigues et al. (2013) recently presented a sequential model for this. They estimate annotator competence as latent variables in a CRF model using EM. They evaluate their approach on synthetic and NER data annotated on Mechanical Turk, showing improvements over the MV baselines and the multi-label model by Dredze et al. (2009). The latter do not model annotator reliability but rather model label priors by integrating them into the CRF objective, and re-estimating them during learning. Both require annotators to supply a full sentence, while we use minimal context, which requires less annotator commitment and makes the task more flexible. Unfortunately, we could not run those models on our data due to label incompatibility and the fact that we typically do not have complete sequences annotated by the same annotators.

Mainzer (2011) actually presents an earlier paper on crowdsourcing POS tagging. However, it differs from our approach in several ways. It uses the Penn Treebank tag set to annotate Wikipedia data (which is much more canonical than Twitter) via a Java applet. The applet automatically labels certain categories, and only presents the users with a series of multiple choice questions for the remainder. This is highly effective, as it eliminates some sources of possible disagreement. In contrast, we do not pre-label any tokens, but always present the annotators with all labels.

8 Conclusion

We use crowdsourcing to collect POS annotations with minimal context (five-word windows). While the performance of POS models learned from this data is still slightly below that of models trained on expert annotations, models learned from aggregations approach oracle performance for POS tagging. In general, we find that the use of a dictionary tends to make aggregations more useful, irrespective of aggregation method. For some downstream tasks, models using the aggregated POS tags perform even better than models using expert-annotated tags.

Acknowledgments

We would like to thank the anonymous reviewers for valuable comments and feedback. This research is funded by the ERC Starting Grant LOW-

LANDS No. 313695.

References

- Chris Callison-Burch and Mark Dredze. 2010. Creating Speech and Language Data With Amazon’s Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk*.
- Bob Carpenter. 2008. Multilevel Bayesian models of categorical data annotation. Technical report, LingPipe.
- A. Philip Dawid and Allan M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, pages 20–28.
- Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.
- Leon Derczynski, Alan Ritter, Sam Clark, and Kalina Bontcheva. 2013. Twitter part-of-speech tagging for all: overcoming sparse and noisy data. In *RANLP*.
- Mark Dredze, Partha Pratim Talukdar, and Koby Crammer. 2009. Sequence learning from data with multiple labels. In *ECML/PKDD Workshop on Learning from Multi-Label Data*.
- Jacob Eisenstein. 2013. What to do about bad language on the internet. In *NAACL*.
- Tim Finin, Will Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. 2010. Annotating named entities in Twitter data with crowdsourcing. In *NAACL-HLT 2010 Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk*.
- Jennifer Foster, Ozlem Cetinoglu, Joachim Wagner, Josef Le Roux, Joakim Nivre, Deirde Hogan, and Josef van Genabith. 2011. From news to comments: Resources and benchmarks for parsing the language of Web 2.0. In *IJCNLP*.
- Kevin Gimpel, Nathan Schneider, Brendan O’Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In *ACL*.
- Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *NAACL*.
- Dirk Hovy, Barbara Plank, and Anders Søgaard. 2014. When pos datasets don’t add up: Combatting sample bias. In *LREC*.

- Mukund Jha, Jacob Andreas, Kapil Thadani, Sara Rosenthal, and Kathleen McKeown. 2010. Corpus creation for new genres: A crowdsourced approach to pp attachment. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk*. Association for Computational Linguistics.
- Mark Johnson. 2007. Why doesn’t EM find good HMM POS-taggers. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *ICML*.
- Shen Li, João Graça, and Ben Taskar. 2012. Wiki-ly supervised part-of-speech tagging. In *EMNLP*.
- Jacob Emil Mainzer. 2011. Labeling parts of speech using untrained annotators on mechanical turk. Master’s thesis, The Ohio State University.
- Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational linguistics*, 20(2):155–171.
- Olutobi Owoputi, Brendan O’Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *NAACL*.
- Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *LREC*.
- Sujith Ravi and Kevin Knight. 2009. Minimized Models for Unsupervised Part-of-Speech Tagging. 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*. Association for Computational Linguistics.
- Vikas C. Raykar and Shipeng Yu. 2012. Eliminating Spammers and Ranking Annotators for Crowdsourced Labeling Tasks. *Journal of Machine Learning Research*, 13:491–518.
- Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *EMNLP*.
- Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. 2013. Sequence labeling with multiple annotators. *Machine Learning*, pages 1–17.
- Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *NAACL*.
- Padhraic Smyth, Usama Fayyad, Mike Burl, Pietro Perona, and Pierre Baldi. 1995. Inferring ground truth from subjective labelling of Venus images. *Advances in neural information processing systems*, pages 1085–1092.
- Rion Snow, Brendan O’Connor, Dan Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good? Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.
- Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. *TACL*, Mar(1):1–12.
- Peter Welinder, Steve Branson, Serge Belongie, and Pietro Perona. 2010. The multidimensional wisdom of crowds. In *NIPS*.
- Jacob Whitehill, Paul Ruvolo, Tingfan Wu, Jacob Bergsma, and Javier Movellan. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in Neural Information Processing Systems*, 22:2035–2043.
- Yan Yan, Rómer Rosales, Glenn Fung, Mark Schmidt, Gerardo Hermosillo, Luca Bogoni, Linda Moy, and Jennifer Dy. 2010. Modeling annotator expertise: Learning when everybody knows a bit of something. In *International Conference on Artificial Intelligence and Statistics*.

Building Sentiment Lexicons for All Major Languages

Yanqing Chen

Computer Science Dept.
Stony Brook University
Stony Brook, NY 11794

`cyanqing@cs.stonybrook.edu`

Steven Skiena

Computer Science Dept.
Stony Brook University
Stony Brook, NY 11794

`skiena@cs.stonybrook.edu`

Abstract

Sentiment analysis in a multilingual world remains a challenging problem, because developing language-specific sentiment lexicons is an extremely resource-intensive process. Such lexicons remain a scarce resource for most languages.

In this paper, we address this lexicon gap by building high-quality sentiment lexicons for 136 major languages. We integrate a variety of linguistic resources to produce an immense knowledge graph. By appropriately propagating from seed words, we construct sentiment lexicons for each component language of our graph. Our lexicons have a polarity agreement of 95.7% with published lexicons, while achieving an overall coverage of 45.2%.

We demonstrate the performance of our lexicons in an extrinsic analysis of 2,000 distinct historical figures' Wikipedia articles on 30 languages. Despite cultural difference and the intended neutrality of Wikipedia articles, our lexicons show an average sentiment correlation of 0.28 across all language pairs.

1 Introduction

Sentiment analysis of English texts has become a large and active research area, with many commercial applications, but the barrier of language limits the ability to assess the sentiment of most of the world's population.

Although several well-regarded sentiment lexicons are available in English (Esuli and Sebastiani, 2006; Liu, 2010), the same is not true for most of the world's languages. Indeed, our literature search identified only 12 *publicly available* sentiment lexicons for only 5 non-English languages (Chinese mandarin, German, Arabic, Japanese and

Italian). No doubt we missed some, but it is clear that these resources are not widely available for most important languages.

In this paper, we strive to produce a comprehensive set of sentiment lexicons for the world's major languages. We make the following contributions:

- *New Sentiment Analysis Resources* – We have generated sentiment lexicons for 136 major languages via graph propagation which are now publicly available¹. We validate our own work through other publicly available, human annotated sentiment lexicons. Indeed, our lexicons have polarity agreement of 95.7% with these published lexicons, plus an overall coverage of 45.2%.
- *Large-Scale Language Knowledge Graph Analysis* – We have created a massive comprehensive knowledge graph of 7 million vocabulary words from 136 languages with over 131 million semantic inter-language links, which proves valuable when doing alignment between definitions in different languages.
- *Extrinsic Evaluation* – We elucidate the sentiment consistency of entities reported in different language editions of Wikipedia using our propagated lexicons. In particular, we pick 30 languages and compute sentiment scores for 2,000 distinct historical figures. Each language pair exhibits a Spearman sentiment correlation of at least 0.14, with an average correlation of 0.28 over all pairs.

The rest of this paper is organized as follows. We review related work in Section 2. In Section 3, we describe our resource processing and design decisions. Section 4 discusses graph propagation methods to identify sentiment polarity across languages. Section 5 evaluates our results against

¹<https://sites.google.com/site/datascienceslab/projects/>

each available human-annotated lexicon. Finally, in Section 6 we present our extrinsic evaluation of sentiment consistency in Wikipedia prior to our conclusions.

2 Related Work

Sentiment analysis is an important area of NLP with a large and growing literature. Excellent surveys of the field include (Liu, 2013; Pang and Lee, 2008), establishing that rich online resources have greatly expanded opportunities for opinion mining and sentiment analysis. Godbole et al. (2007) build up an English lexicon-based sentiment analysis system to evaluate the general reputation of entities. Taboada et al. (2011) present a more sophisticated model by considering patterns, including negation and repetition using adjusted weights. Liu (2010) introduces an efficient method, at the state of the art, for doing sentiment analysis and subjectivity in English.

Researchers have investigated topic or domain dependent approaches to identify opinions. Jijikoun et al. (2010) focus on generating topic specific sentiment lexicons. Li et al. (2010) extract sentiment with global and local topic dependency. Gindl et al. (2010) perform sentiment analysis according to cross-domain contextualization and Pak and Paroubek (2010) focus on Twitter, doing research on colloquial format of English.

Work has been done to generalize sentiment analysis to other languages. Denecke (2008) performs multilingual sentiment analysis using SentiWordNet. Mihalcea et al. (2007) learn multilingual subjectivity via cross-lingual projections. Abbasi et al. (2008) extract specific language features of Arabic which requires language-specific knowledge. Gînscă et al. (2011) work on better sentiment analysis system in Romanian.

The ready availability of machine translation to and from English has prompted efforts to employ translation for sentiment analysis (Bautin et al., 2008). Banea et al. (2008) demonstrate that machine translation can perform quite well when extending the subjectivity analysis to multi-lingual environment, which makes it inspiring to replicate their work on lexicon-based sentiment analysis.

Machine learning approaches to sentiment analysis are attractive, because of the promise of reduced manual processing. Boiy and Moens (2009) conduct machine learning sentiment analysis using multilingual web texts. Deep learning ap-

proaches draft off of distributed word embedding which offer concise features reflecting the semantics of the underlying vocabulary. Turian et al. (2010) create powerful word embedding by training on real and corrupted phrases, optimizing for the replaceability of words. Zou et al. (2013) combine machine translation and word representation to generate bilingual language resources. Socher et al. (2012) demonstrates a powerful approach to English sentiment using word embedding, which can easily be extended to other languages by training on appropriate text corpora.

3 Knowledge Graph Construction

In this section we will describe how we leverage off a variety of NLP resources to construct the semantic connection graph we will use to propagate sentiment lexicons.

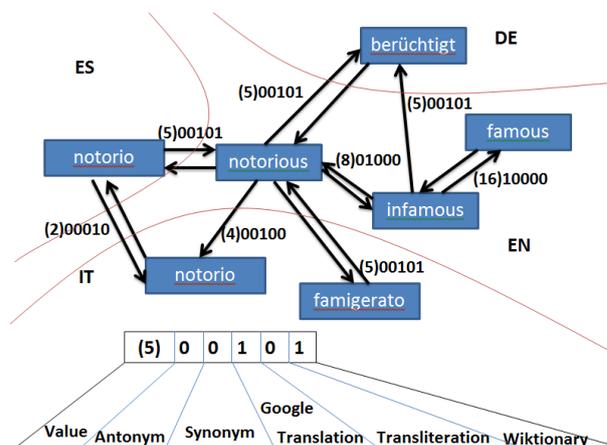


Figure 1: Illustration of our knowledge graph, showing links between words and edge representation to preserve source identity. For each edge between corresponding words, a 5-bit integer will record the existence of 5 possible semantic links.

The Polyglot project (Al-Rfou et al., 2013) identified the 100,000 most frequently used words in each language’s Wikipedia. Drawing a candidate lexicon from Wikipedia has some downsides (e.g. limited use of informal words), but is representative and convenient over a large number of languages. In particular, we collect total of 7,741,544 high-frequency words from 136 languages to serve as vertices in our graph.

We seek to identify as many semantic links across languages as possible to connect our network, and so integrated several resources:

- *Wiktionary* – This growing resource has en-

tries for 171 languages, edited by people with sufficient background knowledge. Wiktionary provides about 19.7% of the total links covering 382,754 vertices in our graph.

- *Machine Translation* - We script the Google translation API to get even more semantic links. In particular we ask for translations of each word in our English vocabulary to 57 languages with available translators as well as going from each known vocabulary word in other languages to English. In total, machine translation provides 53.2% of the total links and establishes connections between 3.5 million vertices.
- *Transliteration Links* - Natural flow brings words across languages with little morphological change. Closely related language pairs (i.e. Russian and Ukrainian) share many characters/words in common. Though not always true, words with same spelling usually have similar meanings so this can improve the coverage of semantic links. Transliteration provides 22.1% of the total links in our experiment.
- *WordNet* - Finally, we gather synonyms and antonyms of English words from WordNet, which prove particularly useful in propagating sentiment across languages. In total we collect over 100,000 pairs of synonyms and antonyms and created 5.0% of the total links.

Links do not always agree in a bidirectional manner, particularly for multi-sense words, thus all links in our network are unidirectional. Figure 1 illustrates how we encode links from different resources in an integer edge value.

4 Graph Propagation

Sentiment propagation starts from English sentiment lexicons. Through semantic links in our knowledge graph, words are able to extend their sentiment polarities to adjacent neighbors. We experimented with both graph propagation algorithm (Velikovich et al., 2010) and label propagation algorithm (Zhu and Ghahramani, 2002; Rao and Ravichandran, 2009). The primary difference between is that label propagation takes multiple paths between two vertices into consideration, while graph propagation utilizes only the best path between word pairs.

We report results from using Liu’s lexicons (Liu, 2010) as seed words. Liu’s lexicons contain 2006 positive words and 4783 negative words. Of these, 1422 positive words and 2956 negative words (roughly 64.5%) appear among the 100,000 English vertices in our graph.

Dataset	Propagation	Acc	Cov
Arabic	Label	0.93	0.45
	Graph	0.94	0.46
German	Label	0.97	0.31
	Graph	0.97	0.32
English	Label	0.92	0.55
	Graph	0.90	0.69
Italian	Label	0.73	0.29
	Graph	0.72	0.32
Japanese	Label	0.57	0.12
	Graph	0.56	0.15
Chinese-1	Label	0.95	0.62
	Graph	0.94	0.65
Chinese-2	Label	0.97	0.70
	Graph	0.97	0.72

Table 1: Graph propagation vs label propagation. *Acc* represents the ratio of identical polarity between our analysis and the published lexicons. *Cov* reflects what fraction of our lexicons overlap with published lexicons.

Our knowledge network is comprised of links from a heterogeneous collection of sources, of different coverage and reliability. For the task of deciding sentiment polarity of words, only antonym links are negative. An edge gains zero weight if both negative and positive links exist. Edges having multiple positive links will be credited the highest weight among all these links. We conducted a grid search on the weight of each type of links to maximize the best overall accuracy on our test data of published non-English sentiment lexicons. To avoid potential overfitting problems, grid search starts from SentiWordNet English lexicons (Esuli and Sebastiani, 2006) instead of Liu’s.

5 Lexicon Evaluation

We collected all available published sentiment lexicons from non-English languages to serve as standard for our evaluation, including Arabic, Italian, German and Chinese. Coupled with English sentiment lexicons provides in total seven different test cases to experiment against, specifically:

Language	lexicon	+/- Ratio	Language	lexicon	+/- Ratio	Language	lexicon	+/- Ratio
Afrikaans	2299	0.40	Albanian	2076	0.41	Amharic	46	0.63
Arabic	2794	0.41	Aragonese	97	0.47	Armenian	1657	0.43
Assamese	493	0.49	Azerbaijani	1979	0.41	Bashkir	19	0.63
Basque	1979	0.40	Belarusian	1526	0.43	Bengali	2393	0.42
Bosnian	2020	0.42	Breton	184	0.42	Bulgarian	2847	0.40
Burmese	461	0.48	Catalan	3204	0.37	Cebuano	56	0.54
Chechen	26	0.65	Chinese	3828	0.34	Chuvash	17	0.76
Croatian	2208	0.40	Czech	2599	0.41	Danish	3340	0.38
Divehi	67	0.67	Dutch	3976	0.38	English	4376	0.32
Esperanto	2604	0.40	Estonian	2105	0.41	Faroese	123	0.43
Finnish	3295	0.40	French	4653	0.35	Frisian	224	0.43
Gaelic	345	0.50	Galician	2714	0.37	German	3974	0.38
Georgian	2202	0.40	Greek	2703	0.39	Gujarati	2145	0.44
Haitian	472	0.44	Hebrew	2533	0.36	Hindi	3640	0.39
Hungarian	3522	0.38	Icelandic	1770	0.40	Ido	183	0.49
Interlingua	326	0.50	Indonesian	2900	0.37	Italian	4491	0.36
Irish	1073	0.45	Japanese	1017	0.39	Javanese	168	0.51
Kazakh	81	0.65	Kannada	2173	0.42	Kirghiz	246	0.49
Khmer	956	0.49	Korean	2118	0.42	Kurdish	145	0.48
Latin	2033	0.46	Latvian	1938	0.42	Limburgish	93	0.46
Lithuanian	2190	0.41	Luxembourg	224	0.52	Macedonian	2965	0.39
Malagasy	48	0.54	Malayalam	393	0.50	Malay	2934	0.39
Maltese	863	0.50	Marathi	1825	0.48	Manx	90	0.51
Mongolian	130	0.52	Nepali	504	0.49	Norwegian	3089	0.37
Nynorsk	1894	0.39	Occitan	429	0.40	Oriya	360	0.51
Ossetic	12	0.67	Panjabi	79	0.63	Pashto	198	0.50
Persian	2477	0.39	Polish	3533	0.39	Portuguese	3953	0.35
Quechua	47	0.55	Romansh	116	0.48	Romanian	3329	0.39
Russian	2914	0.43	Sanskrit	178	0.59	Sami	24	0.71
Serbian	2034	0.41	Sinhala	1122	0.43	Slovak	2428	0.43
Slovene	2244	0.42	Spanish	4275	0.36	Sundanese	476	0.50
Swahili	1314	0.42	Swedish	3722	0.39	Tamil	2057	0.40
Tagalog	1858	0.44	Tajik	97	0.62	Tatar	76	0.50
Telugu	2523	0.41	Thai	1279	0.51	Tibetan	24	0.63
Turkmen	78	0.56	Turkish	2500	0.39	Uighur	18	0.44
Ukrainian	2827	0.41	Urdu	1347	0.39	Uzbek	111	0.57
Vietnamese	1016	0.38	Volapuk	43	0.70	Walloon	193	0.32
Welsh	1647	0.42	Yiddish	395	0.43	Yoruba	276	0.50

Table 2: Sentiment lexicon statistics. We tag 10 languages having most/least sentiment words with blue/green color and 10 languages having highest/lowest ratio of positive words with orange/purple color.

- *Arabic*: (Abdul-Mageed et al., 2011).
- *German*: (Remus et al., 2010).
- *English*: (Esuli and Sebastiani, 2006).
- *Italian*: (Basile and Nissim, 2013).
- *Japanese*: (Kaji and Kitsuregawa, 2007).
- *Chinese-1, Chinese-2*: (He et al., 2010).

We present the accuracy and coverage achieved by two propagation model in Table 1. Both models achieve similar accuracy while slightly more words in graph propagation can be verified via published lexicons. Performance is not good on Japanese because of mismatching between our dictionary and the test data.

Table 2 reveals that very sparse sentiment lexicons resulted for a small but notable fraction of

the languages we analyzed. In particular, only 20 languages yielded lexicons of less than 100 words. Without exception, they all have very small available definitions in Wikitionary. By contrast, 48 languages had lexicons with over 2,000 words, another 16 with between 1,000 and 2,000: clearly large enough to perform a meaningful analysis.

6 Extrinsic Evaluation: Consistency of Wikipedia Sentiment

We consider evaluating our lexicons on the consistency of Wikipedia pages about a particular individual person among various languages. As our candidate entities for analysis, we use the Wikipedia pages of 2,000 most significant people as measured in the recent book *Who's Bigger?* (Skiena and Ward, 2013). The sentiment polarity of a page is simply computed by subtracting

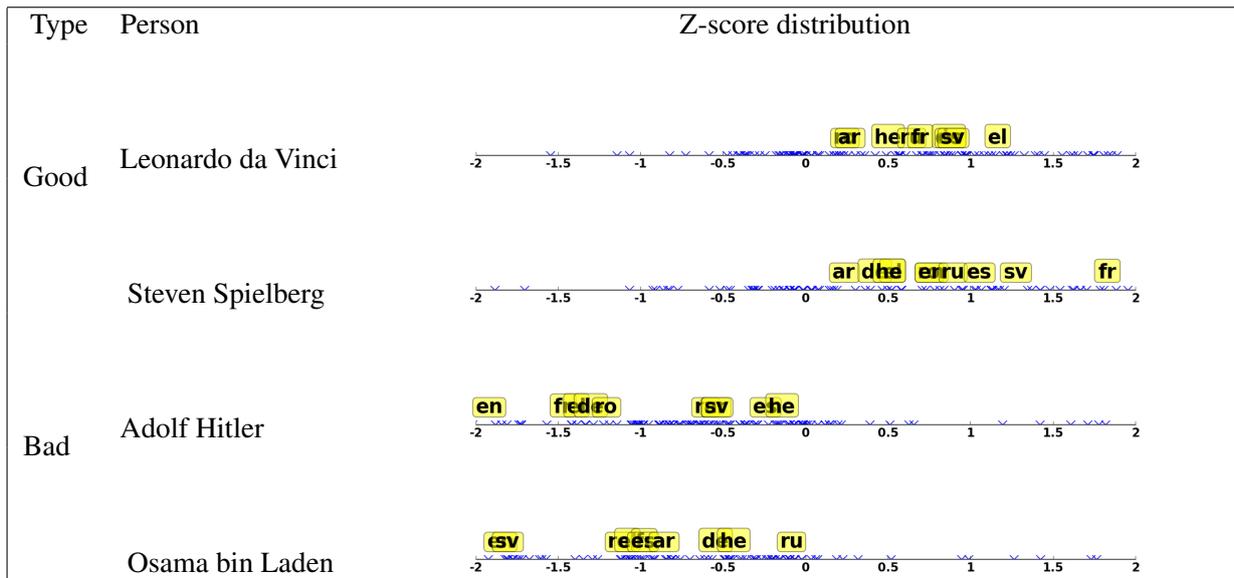


Table 3: Z-score distribution examples. We label 10 languages with their language code and other using tick marks on the x-axis.

the number of negative words from that of positive words, divided by the sum of both.

The differing ratio of positive and negative polarity terms in Table 2 means that sentiment cannot be directly compared across languages. For more consistent evaluation we compute the z-score of each entity against the distribution of all its language’s entities.

We use the Spearman correlation coefficient to measure the consistence of sentiment distribution across all entities with pages in a particular language pair. Figure 2 shows the results for 30 languages with largest propagated sentiment lexicon size. All pairs of language exhibit positive correlation (and hence generally stable and consistent sentiment), with an average correlation of 0.28.

Finally, Table 3 illustrates sentiment consistency over all 136 languages (represented by blue tick marks), with the first 10 languages in Figure 2 granted labels. Respected artists like *Steven Spielberg* and *Leonardo da Vinci* show as consistently positive sentiment as notorious figures like *Osama bin Laden* and *Adolf Hitler* are negative.

7 Conclusions

Our knowledge graph propagation is generally effective at producing useful sentiment lexicons. Interestingly, the ratio of positive sentiment words is strongly connected with number of sentiment words – it is noteworthy that English has the smallest ratio of positive lexicon terms. The

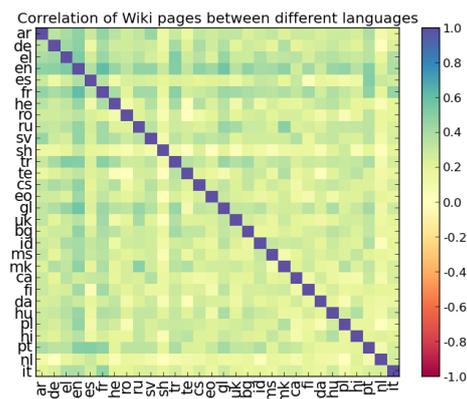


Figure 2: Heatmap of sentiment correlation between 30 languages.

phenomenon possibly shows that many negative words reflecting cultural nuances do not translate well. We believe that this ratio can be considered as quality measurement of the propagation. Similar approaches can be extended to other NLP tasks using different semantic links, specific dictionary and special seed words. Future work will revolve around learning modifiers, negation terms, and various entity/sentiment attribution.

Acknowledgments

This research was partially supported by NSF Grants DBI-1060572 and IIS-1017181, and a Google Faculty Research Award.

References

- Ahmed Abbasi, Hsinchun Chen, and Arab Salem. 2008. Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. *ACM Transactions on Information Systems (TOIS)*, 26(3):12.
- Muhammad Abdul-Mageed, Mona T Diab, and Mohammed Korayem. 2011. Subjectivity and sentiment analysis of modern standard arabic. In *ACL (Short Papers)*, pages 587–591.
- Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual nlp. *arXiv preprint arXiv:1307.1662*.
- Carmen Banea, Rada Mihalcea, Janyce Wiebe, and Samer Hassan. 2008. Multilingual subjectivity analysis using machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 127–135. Association for Computational Linguistics.
- Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on italian tweets. *WASSA 2013*, page 100.
- M. Bautin, L. Vijayarenu, and S. Skiena. 2008. International sentiment analysis for news and blogs. Second Int. Conf. on Weblogs and Social Media (ICWSM 2008).
- Erik Boiy and Marie-Francine Moens. 2009. A machine learning approach to sentiment analysis in multilingual web texts. *Information retrieval*, 12(5):526–558.
- Kerstin Denecke. 2008. Using sentiwordnet for multilingual sentiment analysis. In *Data Engineering Workshop, 2008. ICDEW 2008. IEEE 24th International Conference on*, pages 507–512. IEEE.
- Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422.
- Stefan Gindl, Albert Weichselbraun, and Arno Scharl. 2010. Cross-domain contextualisation of sentiment lexicons. *19th European Conference on Artificial Intelligence (ECAI)*.
- Alexandru-Lucian Gînscă, Emanuela Boroş, Adrian Iftene, Diana TrandabĂţ, Mihai Toader, Marius Corîci, Cenel-Augusto Perez, and Dan Cristea. 2011. Sentimatrix: multilingual sentiment analysis service. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, pages 189–195. Association for Computational Linguistics.
- Namrata Godbole, Manja Srinivasaiah, and Steven Skiena. 2007. Large-scale sentiment analysis for news and blogs. *ICWSM*, 7.
- Yulan He, Harith Alani, and Deyu Zhou. 2010. Exploring english lexicon knowledge for chinese sentiment analysis. *CIPS-SIGHAN Joint Conference on Chinese Language Processing*.
- Valentin Jijkoun, Maarten de Rijke, and Wouter Weerkamp. 2010. Generating focused topic-specific sentiment lexicons. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 585–594. Association for Computational Linguistics.
- Nobuhiro Kaji and Masaru Kitsuregawa. 2007. Building lexicon for sentiment analysis from massive collection of html documents. In *EMNLP-CoNLL*, pages 1075–1083.
- Fangtao Li, Minlie Huang, and Xiaoyan Zhu. 2010. Sentiment analysis with global topics and local dependency. In *AAAI*.
- Bing Liu. 2010. Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2:568.
- Bing Liu. 2013. *Sentiment Analysis and Opinion Mining*. Morgan and Claypool.
- Rada Mihalcea, Carmen Banea, and Janyce Wiebe. 2007. Learning multilingual subjective language via cross-lingual projections. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 976.
- Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *LREC*.
- Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135.
- Delip Rao and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 675–682. Association for Computational Linguistics.
- Robert Remus, Uwe Quasthoff, and Gerhard Heyer. 2010. Sentiws-a publicly available german-language resource for sentiment analysis. In *LREC*.
- Steven Skiena and Charles Ward. 2013. *Who’s Bigger?: Where Historical Figures Really Rank*. Cambridge University Press.
- Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307.

- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.
- Leonid Velikovich, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. 2010. The viability of web-derived polarity lexicons. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 777–785. Association for Computational Linguistics.
- Xiaojin Zhu and Zoubin Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, Carnegie Mellon University.
- Will Y Zou, Richard Socher, Daniel Cer, and Christopher D Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398.

Difficult Cases: From Data to Learning, and Back

Beata Beigman Klebanov*

Educational Testing Service
660 Rosedale Road
Princeton, NJ 08541

bbeigmanklebanov@ets.org

Eyal Beigman*

Liquidnet Holdings Inc.
498 Seventh Avenue
New York, NY 10018

e.beigman@gmail.com

Abstract

This article contributes to the ongoing discussion in the computational linguistics community regarding instances that are difficult to annotate reliably. Is it worthwhile to identify those? What information can be inferred from them regarding the nature of the task? What should be done with them when building supervised machine learning systems? We address these questions in the context of a subjective semantic task. In this setting, we show that the presence of such instances in training data misleads a machine learner into misclassifying clear-cut cases. We also show that considering machine learning outcomes with and without the difficult cases, it is possible to identify specific weaknesses of the problem representation.

1 Introduction

The problem of cases that are difficult for annotation received recent attention from both the theoretical and the applied perspectives. Such items might receive contradictory labels, without a clear way of settling the disagreement. Beigman and Beigman Klebanov (2009) showed theoretically that *hard cases* – items with unreliable annotations – can lead to unfair benchmarking results when found in test data, and, in worst case, to a degradation in a machine learner’s performance on easy, uncontroversial instances if found in the training data. Schwartz et al. (2011) provided an empirical demonstration that the presence of such difficult cases in dependency parsing evaluations

leads to unstable benchmarking results, as different gold standards might provide conflicting annotations for such items. Reidsma and Carletta (2008) demonstrated by simulation that systematic disagreements between annotators negatively impact generalization ability of classifiers built using data from different annotators. Oosten et al. (2011) showed that judgments of readability of the same texts by different groups of experts are sufficiently systematically different to hamper cross-expert generalization of readability classifiers trained on annotations from different groups. Rehbein and Ruppenhofer (2011) discuss the negative impact of systematic simulated annotation inconsistencies on active learning performance on a word-sense disambiguation task.

In this paper, we address the task of classifying words in a text as semantically new or old. Using multiple annotators, we empirically identify instances that show substantial disagreement between annotators. We then discuss those both from the linguistic perspective, identifying some characteristics of such cases, and from the perspective of machine learning, showing that the presence of difficult cases in the training data misleads the machine learner on easy, clear-cut cases – a phenomenon termed *hard case bias* in Beigman and Beigman Klebanov (2009). The main contribution of this paper is in providing additional empirical evidence in support of the argument put forward in the literature regarding the need to pay attention to problematic, disagreeable instances in annotated data – not only from the linguistic perspective, but also from a machine learning one.

2 Data

The task considered here is that of classifying first occurrences of words in a text as semantically old or new. One of goals of the project is to investigate the relationship between various kinds of non-novelty in text, and, in particular, the rela-

¹The work presented in this paper was done when the first author was a post-doctoral fellow at Northwestern University, Evanston, IL and the second author was a visiting assistant professor at Washington University, St. Louis, MO.

tionship between semantic non-novelty (conceptualized as semantic association with some preceding word in the text), the information structure in terms of given and new information, and the cognitive status of discourse entities (Postolache et al., 2005; Birner and Ward, 1998; Gundel et al., 1993; Prince, 1981). If an annotator identified an associative tie from the target word back to some other word in the text, the target word is thereby classified as semantically old (class **1**, or **positive**); if no ties were identified, it is classified as new (class **0**, or **negative**).

For the project, annotations were collected for 10 texts of various genres, where annotators were asked, for every first appearance of a word in a text, to point out previous words in the text that are semantically or associatively related to it. All data was annotated by 22 undergraduate and graduate students in various disciplines who were recruited for the task. During outlier analysis, data from two annotators was excluded from consideration, while 20 annotations were retained. This task is fairly subjective, with inter-annotator agreement $\kappa=0.45$ (Beigman Klebanov and Shamir, 2006).

Table 1 shows the number and proportion of instances that received the “semantically old” (**1**) label from i annotators, for $0 \leq i \leq 20$. The first column shows the number of annotators who gave the label “semantically old” (1). Column 2 shows the number and proportion of instances that received the label 1 from the number of annotators shown in column 1. Column 3 shows the split into item difficulty groups. We note that while about 20% of the instances received a unanimous **0** annotation and about 12% of the instances received just one **1** label out of 20 annotators, the remaining instances are spread out across various values of i . Reasons for this spread include intrinsic difficulty of some of the items, as well as attention slips. Since annotators need to consider the whole of the preceding text when annotating a given word, maintaining focus is a challenge, especially for words that first appear late in the text.

Our interest being in difficult, disagreeable cases, we group the instances into 5 bands according to the observed level of disagreement and the tendency in the majority of the annotations. Thus, items with at most two label **1** annotations are clearly semantically new, while those with at least 17 (out of 20) are clearly semantically old. The groups *Hard 0* and *Hard 1* contain instances

# 1s	# instances (proportion)	group
0	476 (.20)	Easy 0 (.40)
1	271 (.12)	
2	191 (.08)	
3	131 (.06)	Hard 0 (.25)
4	106 (.05)	
5	76 (.03)	
6	95 (.04)	
7	85 (.04)	
8	78 (.03)	
9	60 (.03)	Very Hard (.08)
10	70 (.03)	
11	60 (.03)	
12	57 (.02)	Hard 1 (.13)
13	63 (.03)	
14	68 (.03)	
15	49 (.02)	
16	65 (.03)	
17	60 (.03)	Easy 1 (.14)
18	72 (.03)	
19	94 (.04)	
20	99 (.04)	

Table 1: Sizes of subsets by levels of agreement.

with at least a 60% majority classification, while the middle class – *Very Hard* – contains instances for which it does not appear possible to even identify the overall tendency.

In what follows, we investigate the learnability of the classification of semantic novelty from various combinations of easy, hard, and very hard data.

3 Experimental Setup

3.1 Training Partitions

The objective of the study is to determine the usefulness of instances of various types in the training data for semantic novelty classification. In particular, in light of Beigman and Beigman Klebanov (2009), we want to check whether the presence of less reliable data (hard cases) in the training set adversely impacts performance on the highly reliable data (easy cases). We therefore test separately on easy and hard cases.

We ran 25 rounds of the following experiment. All easy cases are randomly split 80% (train) and 20% (test), all hard cases are split into train and test sets in the same proportions. Then various

parts of the training data are used to train the 5 systems described in Table 2. We build models using easy data; hard data; easy and hard data; easy, hard, and very hard data; easy data and a weighted sample of the hard data. The labels for very hard data were assigned by flipping a fair coin.

System	Easy	Hard	Very Hard
E	+		
H		+	
E+H	+	+	
E+H+VH	+	+	+
E+H _w ¹⁰⁰	+	sample ¹	

Table 2: The 5 training regimes used in the experiment, according to the parts of the data utilized for training.

3.2 Machine Learning

We use linear Support Vector Machines classifier as implemented in SVMLight (Joachims, 1999). Apart from being a popular and powerful machine learning method, linear SVM is one of the family of classifiers analyzed in Beigman and Beigman Klebanov (2009), where they are theoretically shown to be vulnerable to hard case bias in the worst case.

To represent the instances, we use two features that capture semantic relatedness between words. One feature uses Latent Semantic Analysis (Deerwester et al., 1990) trained on the Wall Street Journal articles to quantify the distributional similarity of two words, the other uses an algorithm based on WordNet (Miller, 1990) to calculate semantic relatedness, combining information from both the hierarchy and the glosses (Beigman Klebanov, 2006). For each word, we calculate LSA (WordNet) relatedness score for this word with each preceding word in the text, and report the highest pairwise score as the LSA (WordNet) feature value for the given word. The values of the features can be thought of as quantifying the strength of the evidence for semantic non-newness that could be obtained via a distributional or a dictionary-based method.

¹The weight corresponds to the number of people who marked the item as 1, for hard cases. We take a weighted sample of 100 hard cases.

4 Results

We calculate the accuracy of every system separately on the easy and hard test data. Table 3 shows the results.

Train	Test-E		Test-H	
	Acc	Rank	Acc	Rank
E	0.781	1	0.643	2
E+H	0.764	2	0.654	1
E+H+VH	0.761	2	0.650	1,2
H	0.620	3	0.626	3
E+H _w ¹⁰⁰	0.779	1	0.645	2

Table 3: Accuracy and ranking for semantic novelty classification for systems built using various training data and tested on easy (Test-E) and hard (Test-H) cases. Systems with insignificant differences in performance (paired t-test, n=25, p>0.05) are given the same rank.

We observe first the performance of the system trained *solely* on hard cases (H in Table 3). This system shows the worst performance, both on the easy test and on the hard test. In fact, this system failed to learn anything about the positive class in 24 out of the 25 runs, classifying all cases as negative. It is thus safe to conclude that in the feature space used here the supervision signal in the hard cases is too weak to guide learning.

The system trained *solely* on easy cases (E in Table 3) significantly outperforms H both on the easy and on the hard test. That is, easy cases are *more* informative about the classification of hard cases than the hard cases themselves. This shows that at least some hard cases pattern similarly to the easy ones in the feature space; SVM failed to single them out when trained on hard cases alone, but they are learnable from the easy data.

The system that trained on all cases – both easy and hard – attains the best performance on hard cases but yields to E on the easy test (Test-E). This demonstrates what Beigman and Beigman Klebanov (2009) called *hard case bias* – degradation in test performance on easy cases due to hard cases in the training data. The negative effect of using hard cases in training data can be mitigated if we only use a small sample of them (system E+H_w¹⁰⁰); yet neither this nor other schemes we tried of selectively incorporating hard cases into training data produced an improvement over E when tested on easy cases (Test-E).

5 Discussion

5.1 Beyond worst case

Beigman and Beigman Klebanov (2009) performed a theoretical analysis showing that hard cases could lead to hard case bias where hard cases have completely un-informative labels, with probability of $p=0.5$ for either label. These correspond to very hard cases in our setting. According to Table 3, it is indeed the case that adding the very hard cases hurts performance, but not significantly so – compare results for E+H vs E+H+VH systems.

Our results suggest that un-informative labels are not necessary for the hard case bias to surface. The instances grouped under Hard 1 have the probability of $p=0.66$ for class 1 and the instances grouped under Hard 0 have the probability of $p=0.71$ for class 0. Thus, while the labels are somewhat informative, it is apparently the case that the hard instances are distributed sufficiently differently in the feature space from the easy cases with the same label to produce a hard case bias.

Inspecting the distribution of hard cases (Figure 1), we note that hard cases do not follow the worst case pattern analyzed in Beigman and Beigman Klebanov (2009), where they were concentrated in an area of the feature space that was removed far from the separation plane, a malignant but arguably unlikely scenario (Dligach et al., 2010). Here, hard cases are spread both close and far from the plane, yet their distribution is sufficiently different from that of the easy cases to produce hard case bias during learning.

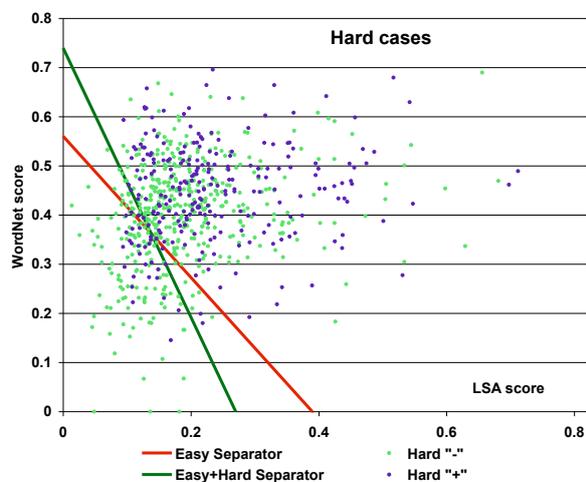


Figure 1: Hard cases with separators learned from easy and easy+hard training data.

5.2 The nature of hard cases

Figure 1 plots the hard instances in the two-dimensional feature space: Latent Semantic Analysis score is shown on x-axis, and WordNet-based score is shown on the y-axis. The red lines show the linear separator induced when the system is trained on easy cases only (system E in Table 3), whereas the green line shows the separator induced when the system is trained on both easy and hard cases (system E+H).

It is apparent from the figure that the difference in the distributions of the easy and the hard cases lead to a lower threshold for LSA score when WordNet score is zero and a higher threshold of WordNet score when LSA score is zero in hard vs easy cases. That is, the system exposed to hard cases learned to trust LSA more and to trust WordNet less when determining that an instance is semantically old than a system that saw only easy cases at train time.

The tendency to trust WordNet less yields an improvement in precision (92.1% for system E+H on Test-E class 1 data vs 84% for system E on Test-E class 1 data), which comes at a cost of a drop in recall (42.2% vs 53.3%) on easy positive cases. This suggests that high WordNet scores that are not supported by distributional evidence are a source of Hard 0 cases that made the system more cautious when relying on WordNet scores.

The pattern of low LSA score and high WordNet score often obtains for rare senses of words: Distributional evidence typically points away from these senses, but they can be recovered through dictionary definitions (glosses) in WordNet.

An example of hard 0 case involves a homonymous rare sense. *Deck* is used in the *observation deck* sense in one of the texts. However, it was found to be highly related to *buy* by WordNet-based measure through the notion of *illegal – buy* in the sense of *bribe* and *deck* in the sense of *a packet of illegal drugs*. This is clearly a spurious connection that makes *deck* appear semantically associated with preceding material, whereas annotators largely perceived it as new.

Exposure to such cases at training time leads the system to forgo handling rare senses that lack distributional evidence, thus leading to misclassification of easy positive cases that exhibit a similar pattern. Thus, *stall* and *market* are both used in the sales outlet sense in one of the text. They come out highly related by WordNet measure; yet in the 68

instances of *stall* in the training data for LSA the homonymous verbal usage predominates. Similarly, *partner* is overwhelmingly used in the *business partner* sense in the WSJ data, hence *wife* and *partner* come out distributionally unrelated, while the WordNet based measure successfully recovers these connections.

Our features, while rich enough to diagnose a rare sense (low LSA score and high WordNet score), do not provide information regarding the appropriateness of the rare sense in context. Short of full scale word sense disambiguation, we experimented with the idea of taking the *second* highest pairwise score as the value of the WordNet feature, under the assumption that an appropriate rare sense is likely to be related to multiple words in the preceding text, while a spurious rare sense is less likely to be accidentally related to more than one preceding word. We failed to improve performance, however; it is thus left for future work to enrich the representation of the problem so that cases with inappropriate rare senses can be differentiated from the appropriate ones. In the context of the current article, the identification of a particular weakness in the representation is an added value of the analysis of the machine learning performance with and without the difficult cases.

6 Related Work

Reliability of annotation is a concern widely discussed in the computational linguistics literature (Bayerl and Paul, 2011; Beigman Klebanov and Beigman, 2009; Artstein and Poesio, 2008; Craggs and McGee Wood, 2005; Di Eugenio and Glass, 2004; Carletta, 1996). Ensuring high reliability is not always feasible, however; the advent of crowdsourcing brought about interest in algorithms for recovering from noisy annotations: Snow et al. (2008), Passonneau and Carpenter (2013) and Raykar et al. (2010) discuss methods for improving over annotator majority vote when estimating the ground truth from multiple noisy annotations.

A situation where learning from a small number of carefully chosen examples leads to a better performance in classifiers is discussed in the active learning literature (Schohn and Cohn, 2000; Cebon and Berthold, 2009; Nguyen and Smeulders, 2004; Tong and Koller, 2001). Recent work in the *proactive* active learning and *multi-expert* active learning paradigms incorporates considera-

tions of item difficulty and annotator expertise into an active learning scheme (Wallace et al., 2011; Donmez and Carbonell, 2008).

In information retrieval, one line of work concerns the design of evaluation schemes that reflect different levels of document relevance to a given query (Kanoulas and Aslam, 2009; Sakai, 2007; Kekäläinen, 2005; Sormunen, 2002; Voorhees, 2001; Järvelin and Kekäläinen, 2000; Voorhees, 2000). Järvelin and Kekäläinen (2000) consider, for example, a tiered evaluation scheme, where precision and recall are reported separately for every level of relevance, which is quite analogous to the idea of testing separately on easy and hard cases as employed here. The graded notion of relevance addressed in the information retrieval research assumes a coding scheme where people assign documents into one of the relevance tiers (Kekäläinen, 2005; Sormunen, 2002). In our case, the graded notion of semantic novelty is a possible explanation for the observed pattern of annotator responses.

7 Conclusion

This article contributes to the ongoing discussion in the computational linguistics community regarding instances that are difficult to annotate reliably – how to identify those, and what to do with them once identified. We addressed this issue in the context of a subjective semantic task. In this setting, we showed that the presence of difficult instances in training data misleads a machine learner into misclassifying clear-cut, easy cases. We also showed that considering machine learning outcomes with and without the difficult cases, it is possible to identify specific weaknesses of the problem representation. Our results align with the literature suggesting that difficult cases in training data can be disruptive (Beigman and Beigman Klebanov, 2009; Schwartz et al., 2011; Rehbein and Ruppenhofer, 2011; Reidsma and Carletta, 2008); yet we also show that investigating their impact on the learning outcomes in some detail can provide insight about the task at hand.

The main contribution of this paper is therefore in providing additional empirical evidence in support of the argument put forward in the literature regarding the need to pay attention to problematic, disagreeable instances in annotated data – both from the linguistic and from the machine learning perspectives.

References

- Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.
- Petra Saskia Bayerl and Karsten Ingmar Paul. 2011. What determines inter-coder agreement in manual annotations? a meta-analytic investigation. *Comput. Linguist.*, 37(4):699–725, December.
- Eyal Beigman and Beata Beigman Klebanov. 2009. Learning with Annotation Noise. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics*, pages 280–287, Singapore, August.
- Beata Beigman Klebanov and Eyal Beigman. 2009. From Annotator Agreement to Noise Models. *Computational Linguistics*, 35(4):493–503.
- Beata Beigman Klebanov and Eli Shamir. 2006. Reader-based exploration of lexical cohesion. *Language Resources and Evaluation*, 40(2):109–126.
- Beata Beigman Klebanov. 2006. Measuring Semantic Relatedness Using People and WordNet. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 13–16, New York City, USA, June. Association for Computational Linguistics.
- Betty Birner and Gregory Ward. 1998. *Information Status and Non-canonical Word Order in English*. Amsterdam/Philadelphia: John Benjamins.
- Jean Carletta. 1996. Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254.
- Nicolas Cebtron and Michael Berthold. 2009. Active learning for object classification: From exploration to exploitation. *Data Mining and Knowledge Discovery*, 18:283–299.
- Richard Craggs and Mary McGee Wood. 2005. Evaluating Discourse and Dialogue Coding Schemes. *Computational Linguistics*, 31(3):289–296.
- Scott Deerwester, Susan Dumais, George Furnas, Thomas Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society For Information Science*, 41:391–407.
- Barbara Di Eugenio and Michael Glass. 2004. The kappa statistic: a second look. *Computational Linguistics*, 30(1):95–101.
- Dmitriy Dligach, Rodney Nielsen, and Martha Palmer. 2010. To Annotate More Accurately or to Annotate More. In *Proceedings of the 4th Linguistic Annotation Workshop*, pages 64–72, Uppsala, Sweden, July.
- Pinar Donmez and Jaime G. Carbonell. 2008. Proactive learning: Cost-sensitive active learning with multiple imperfect oracles. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM '08*, pages 619–628, New York, NY, USA. ACM.
- Jeanette Gundel, Nancy Hedberg, and Ron Zacharski. 1993. Cognitive status and the form of referring expressions in discourse. *Language*, 69:274–307.
- Kalervo Järvelin and Jaana Kekäläinen. 2000. IR Evaluation Methods for Retrieving Highly Relevant Documents. In *Proceedings of the 23th Annual International Conference on Research and Development in Information Retrieval*, pages 41–48, Athens, Greece, July.
- Thorsten Joachims. 1999. Advances in Kernel Methods - Support Vector Learning. In Bernhard Schölkopf, Christopher Burges, and Alexander Smola, editors, *Making large-scale SVM learning practical*, pages 169–184. MIT Press.
- Evangelos Kanoulas and Javed Aslam. 2009. Empirical Justification of the Gain and Discount Function for nDCG. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management*, pages 611–620, Hong Kong, November.
- Jaana Kekäläinen. 2005. Binary and graded relevance in IR evaluations – Comparison of the effects on ranking of IR systems. *Information Processing and Management*, 41:1019–1033.
- George Miller. 1990. WordNet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–312.
- Hieu Nguyen and Arnold Smeulders. 2004. Active Learning Using Pre-clustering. In *Proceedings of 21st International Conference on Machine Learning*, pages 623–630, Banff, Canada, July.
- Philip Oosten, Vronique Hoste, and Dries Tanghe. 2011. A posteriori agreement as a quality measure for readability prediction systems. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 6609 of *Lecture Notes in Computer Science*, pages 424–435. Springer Berlin Heidelberg.
- Rebecca J. Passonneau and Bob Carpenter. 2013. The benefits of a model of annotation. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 187–195, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Oana Postolache, Ivana Kruijff-Korbayova, and Geert-Jan Kruijff. 2005. Data-driven approaches for information structure identification. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 9–16, Vancouver, British Columbia, Canada, October.

- Ellen Prince. 1981. Toward a taxonomy of given-new information. In Peter Cole, editor, *Radical Pragmatics*, pages 223–255. Academic Press.
- Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *J. Mach. Learn. Res.*, 11:1297–1322, August.
- Ines Rehbein and Josef Ruppenhofer. 2011. Evaluating the impact of coder errors on active learning. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT '11*, pages 43–51, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Dennis Reidsma and Jean Carletta. 2008. Reliability Measurement without Limits. *Computational Linguistics*, 34(3):319–326.
- Tetsuya Sakai. 2007. On the reliability of information retrieval metrics based on graded relevance. *Information Processing and Management*, 43:531–548.
- Greg Schohn and David Cohn. 2000. Less is more: Active Learning with Support Vector Machines. In *Proceedings of 17th International Conference on Machine Learning*, pages 839–846, San Francisco, July.
- Roy Schwartz, Omri Abend, Roi Reichart, and Ari Rappoport. 2011. Neutralizing linguistically problematic annotations in unsupervised dependency parsing evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT '11*, pages 663–672, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Rion Snow, Brendan O’Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08*, pages 254–263, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Eero Sormunen. 2002. Liberal relevance criteria of TREC – Counting on negligible documents? In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 324–330, Tampere, Finland, August.
- Simon Tong and Daphne Koller. 2001. Support Vector Machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66.
- Ellen Voorhees. 2000. Variations in relevance judgments and the measurement of retrieval effectiveness. *Information Processing and Management*, 36:697–716.
- Ellen Voorhees. 2001. Evaluation by highly relevant documents. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 74–82, New Orleans, LA, USA, September.
- B. Wallace, K. Small, C. Brodley, and T. Trikalinos, 2011. *Who Should Label What? Instance Allocation in Multiple Expert Active Learning*, chapter 16, pages 176–187.

The VerbCorner Project: Findings from Phase 1 of Crowd-Sourcing a Semantic Decomposition of Verbs

Joshua K. Hartshorne

Department of Brain and Cognitive Sciences
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139, USA
jkhartshorne@gmail.com

Claire Bonial, Martha Palmer

Department of Linguistics
University of Colorado at Boulder
Hellems 290, 295 UCB
Boulder, CO 80309, USA
{CBonial, MPalmer}@colorado.edu

Abstract

Any given verb can appear in some syntactic frames (*Sally broke the vase*, *The vase broke*) but not others (**Sally broke at the vase*, **Sally broke the vase to John*). There is now considerable evidence that the syntactic behaviors of some verbs can be predicted by their meanings, and many current theories posit that this is true for most if not all verbs. If true, this fact would have striking implications for theories and models of language acquisition, as well as numerous applications in natural language processing. However, empirical investigations to date have focused on a small number of verbs. We report on early results from VerbCorner, a crowd-sourced project extending this work to a large, representative sample of English verbs.

1 Introduction

Verbs vary in terms of which syntactic frames they can appear in (Table 1). In principle, this could be an unpredictable fact about the verb that must be acquired, much like the phonological form of the verb.

However, most theorists posit that there is a systematic relationship between the semantics of a verb and the syntactic frames in which it can appear (Levin and Hovav, 2005). For instance, it is argued that verbs like *break*, which describe a

caused change of state, can appear in both the NP V NP form (*Sally broke the vase*) and the NP V form (*The vase broke*). Verbs such as *hit* and *like* do not describe a change of state and so cannot appear in both forms.¹ Similarly, only verbs that describe propositional attitudes, such as *like*, can take a *that* complement (*John liked that Sally broke the vase*).

1.1 The Semantic Consistency Hypothesis

This account has a natural consequence, which we dub the Semantic Consistency Hypothesis: There is some set of semantic features such that verbs that share the same syntactic behavior are identical along those semantic features.² Note that on certain accounts, this is a strong tendency rather than a strict necessity (e.g., Goldberg, 1995).

It is widely recognized that a principled relationship between syntax and semantics would have broad implications. It is frequently invoked in theories of language acquisition. For instance, Pinker (1984, 1989) has described how this correspondence could solve long-standing puzzles about how children learn syntax in the first place. Conversely, Gleitman (1990) has shown such a syntax-semantics relationship could solve significant problems in vocabulary acquisition. In fact, both researchers argue that a principled relationship between syntax and semantics is necessary for language to be learnable at all.

In computational linguistics and natural language processing, some form of the Semantic Consistency Hypothesis is often included in linguistic resources and utilized in applications. We

Frame	hit	like	break
NP V NP	x	x	x
NP V	-	-	x
NP that S	-	x	-
NP V at NP	x	-	-

Table 1: Some of the syntactic frames available for *hit*, *like*, and *break*.

¹Note that this is a simplification in that there are non-causal verbs that appear in both the NP V NP frame and the NP V frame. For details, see (Levin, 1993).

²There is a long tradition of partitioning semantics into those aspects of meaning which are “grammatically relevant” and those which are not. We refer the interested reader to Pinker (1989), Jackendoff (1990), and Levin & Rappaport Hovav (2005).

describe in detail one such resource, VerbNet, which is highly relevant to our investigation.

1.2 VerbNet

VerbNet (Kipper et al., 2008; based on Levin, 1993) lists over 6,000 verbs, categorized into 280 classes according to the syntactic frames they can appear in. That is, all verbs in the same class appear in the same set of syntactic frames. Importantly, in addition to characterizing the syntactic frames associated with each class, VerbNet also characterizes the semantics of each class.

For instance, class 9.7, which comprises a couple dozen verbs, allows 7 different syntactic frames. The entry for one frame is shown below:

Syntactic Frame NP V NP PP.DESTINATION

Example Jessica sprayed the wall.

Syntax AGENT V THEME {+LOC|+DEST_CONF}
DESTINATION

Semantics MOTION(DURING(E), THEME)

NOT(PREP(START(E), THEME, DESTINATION))

PREP(END(E), THEME, DESTINATION)

CAUSE(AGENT, E)

Importantly, the semantics listed here is not just for the verb *spray* but applies to all verbs from the Spray Class whenever they appear in that syntactic frame – that is, VerbNet assumes the Semantic Consistency Hypothesis.

VerbNet and its semantic features have been used in a variety of NLP applications, such as semantic role labeling (Swier and Stevenson, 2004), inferencing (Zaenen et al., 2008), verb classification (Joanis et al., 2008), and information extraction (Maynard et al., 2009). It has also been employed in models of language acquisition (Parisien and Stevenson, 2011; Barak et al., 2012). In general, there has been interest in the NLP literature in using these syntactically-relevant semantic features for shallow semantic parsing (e.g., Giuglea and Moschitti, 2006).

2 Empirical Status of the Semantic Consistency Hypothesis

Given the prominence of the Semantic Consistency Hypothesis in both theory and practice, one might expect that it was on firm empirical footing. That is, ideally there would be some database of semantic judgments for a comprehensive set of verbs from each syntactic class. In princi-

ple, these judgments would come from naive annotators, since researchers' intuitions about subtle judgments may be unconsciously clouded by theoretical commitments (Gibson and Fedorenko, 2013). The Semantic Consistency Hypothesis would be supported if, within that database, predicates with the same syntactic properties were systematically related semantically.

No such database exists, whether consisting of the judgments of linguists or naive annotators. Most theoretical studies report researcher judgments for only a handful of examples; how many additional examples were considered by the researcher goes unreported. In any case, to our knowledge, of the 280 syntactic verb classes listed by VerbNet, only a handful have been studied in any detail.

The strongest evidence comes from experimental work on several so-called alternations (the passive, causative, locative, and dative alternations). Here, there does appear to be a systematic semantic distinction between the two syntactic frames in each alternation, at least most of the time. This has been tested with a reasonable sample of the relevant verbs and also in both children and adults (Ambridge et al., 2013; Pinker, 1989). However, the relevant verbs make up a tiny fraction of all English verbs, and even for these verbs, the syntactic frames in question represent only a fraction of the syntactic frames available to those verbs.

This is not an accidental oversight. The limiting factor is scale: with many thousands of verbs and over a hundred commonly-discussed semantic features and syntactic frames, it is not feasible for a single researcher, or even team of researchers, to check which verbs appear in which syntactic frames and carry which semantic entailments. Collecting data from naive subjects is even more laborious, particularly since the average Man on the Street is not necessarily equipped with metalinguistic concepts like *caused change of state* and *propositional attitude*. The VerbCorner Project is aimed at filling that empirical gap.

3 VerbCorner

The VerbCorner Project³ is devoted to collecting semantic judgments for a comprehensive set of verbs along a comprehensive set of theoretically-relevant semantic dimension. These data can be used to test the Semantic Consistency Hypothesis.

³<http://gameswithwords.org/VerbCorner/>

Independent of the validity of that hypothesis, the semantic judgments themselves should prove useful for any study of linguistic meaning or related application.

We address the issue of scale through crowd-sourcing: Recruiting large numbers of volunteers, each of whom may provide only a few annotations. Several previous projects have successfully crowd-sourced linguistic annotations, such as Phrase Detectives, where volunteers have contributed 2.5 million judgments on anaphoric relations (Poesio et al., 2012).

3.1 Integration with VerbNet

One significant challenge for any such project is first classifying verbs according to the syntactic frames they can appear in. Thus, at least initially, we are focusing on the 6,000+ verbs already cataloged in VerbNet. As such, the VerbCorner Project is also verifying and validating the semantics currently encoded in VerbNet. VerbNet will be edited as necessary based on the empirical results.

Integration with VerbNet has additional benefits, since VerbNet itself is integrated with a variety of linguistic resources, such as PropBank and Penn TreeBank. This amplifies the impact of any VerbCorner-inspired changes to VerbNet.

3.2 The Tasks

We selected semantic features of interest based on those most commonly cited in the linguistics literature, with a particular focus on those that – according to VerbNet – apply to many predicates.

Previous research has shown that humans find it easier to reason about real-world scenarios than make abstract judgments (Cosmides and Tooby, 1992). Thus, for each feature (e.g., MOVEMENT), we converted the metalinguistic judgment (“Does this verb entail movement on the part of some entity?”) into a real-world problem.

For example, in “Simon Says Freeze,” a task designed to elicit judgments about movement, the Galactic Overlord (Simon) decrees “Galactic Stay Where You Are Day,” during which nobody is allowed to move from their current location. Participants read descriptions of events and decide whether anyone violated the rule.

In “Explode on Contact,” designed to elicit judgments about physical contact, objects and people explode when they touch one another. The participant reads descriptions of events and decides whether anything has exploded.

Note that each task is designed to elicit judgments about entailments – things that must be true rather than are merely likely to be true. If John greeted Bill, they might have come into contact (e.g., by shaking hands), but perhaps they did not. Previous work suggests that it is the semantic *entailments* that matter, particularly for explaining the syntactic behavior of verbs (Levin, 1993).

3.3 The Items

The exact semantics associated with a verb may depend on its syntactic frame. Thus *Sally rolled the ball* entails that somebody applied force to the ball (namely: Sally), whereas *The ball rolled* does not. Thus, we investigate the semantics of each verb in each syntactic frame available to it (as described by VerbNet). Below, the term *item* is the unit of annotation: a verb in a frame.

In order to minimize unwanted effects of world knowledge, the verb’s arguments are replaced with nonsense words or randomly chosen proper names (*Sally sprayed the dax onto the blicket*). The use of novel words is explained by the story for each task.

3.4 The Phases

Given the sheer scale of the project, data-collection is expected to take several years at least. Thus, data-collection has been broken up into a series of phases. Each phase focuses on a small number of classes and/or semantic entailments. This ensures that there are meaningful intermediate results that can be disseminated prior to the completion of the entire project. This manuscript reports the results of Phase 1.

4 Results

The full data and annotations will be released in the near future and may be available now by request. Below, we summarize the main findings thus far.

4.1 Description of Phase 1

In Phase 1 of the project, we focused on 11 verb classes (Table 3) comprising 641 verbs and seven different semantic entailments (Table 2). While six of these entailments were chosen from among those features widely believed to be relevant for syntax, one was not: A Good World, which investigated evaluation (*Is the event described by the verb positive or negative?*). Although evaluation

Task	Semantic Feature	Anns.	Anns./Item	Mode	Consistency
Entropy	PHYSICAL CHANGE	23,875	7	86%	95%
Equilibrium	APPLICATION OF FORCE	27,128	8	79%	95%
Explode on Contact	PHYSICAL CONTACT	23,590	7	93%	95%
Fickle Folk	CHANGE OF MENTAL STATE	16,466	5	81%	96%
Philosophical Zombie Hunter	MENTAL STATE	24,592	7	80%	89%
Simon Says Freeze	LOCATION CHANGE	24,245	7	83%	88%
A Good World	EVALUATION	22,668	7	72%	74%

Table 2: Respectively: Task, semantic feature tested, number of annotations, mean number of annotations per item, mean percentage of participants choosing the modal response, consistency within class.

of events is an important component of human psychology, to our knowledge no researcher has suggested that it is relevant for syntax. As such, this task provides a lower bound for how much semantic consistency one might expect within a syntactic verb class.

In all, we collected 162,564 judgments from 1,983 volunteers (Table 2).

4.2 Inter-annotator Agreement

Each task had been iteratively piloted and redesigned until inter-annotator reliability was acceptable, as described in a previous publication. However, these pilot studies involved a small number of items which were coded by all annotators. How good was the reliability in the crowdsourcing context?

Because we recruited large numbers of annotators, most of whom annotated only a few items, typical measures of inter-annotator agreement such as Cohen’s *kappa* are not easily calculated. Instead, for each item, we calculated the most common (modal) response. We then con-

sidered what proportion of all annotations were accounted for by the modal response: a mean of 100% would indicate that there was no disagreement among annotators for any item.

As can be seen in Table 2, for every task, the modal response covered the bulk responses, ranging from a low of 72% for EVALUATION to a high of 93% for PHYSICAL CONTACT. Since there were typically 4 or more possible answers per item, inter-annotator agreement was well above chance. This represents good performance given that the annotators were entirely untrained.

In many cases, annotator disagreement seems to be driven by syntactic constructions that are only marginally grammatical. For instance, inter-annotator agreement was typically low for class 63. VerbNet suggests two syntactic frames for class 63, one of which (NP V THAT S) appears to be marginal (*?I control that Mary eats*). In fact, annotators frequently flagged these items as ungrammatical, which is a valuable result in itself for improving VerbNet.

Class	Examples	PChange	Force	Contact	MChange	Mental	LChange
12	yank, press	-	x	d	-	-	d
18.1	hit, squash	d	x	d	-	-	d
29.5	believe, conjecture	-	-	-	-	d	-
31.1	amuse, frighten	-	-	-	x	d	-
31.2	like, fear	-	-	-	-	x	-
45.1	break, crack	x	d	d	-	-	d
51.3.1	bounce, roll	-	d	d	-	-	d
51.3.2	run, slink	-	d	-	-	-	d
51.6	chase, follow	-	-	-	-	-	d
61	attempt, try	-	-	-	-	-	-
63	control, enforce	-	-	-	-	-	-

Table 3: VerbNet classes investigated in Phase 1, with presence of semantic entailments as indicated by data. *x* = feature present; *-* = feature absent; *d* = depends on syntactic frame.

4.3 Testing the Semantic Consistency Hypothesis

4.3.1 Calculating consistency

We next investigated whether our results support the Semantic Consistency Hypothesis. As noted above, the question is not whether all verbs in the same syntactic class share the same semantic entailments. Even a single verb may have different semantic entailments when placed in different syntactic frames. Thus, calculating consistency of a class must take differing frames into account.

There are many sophisticated rubrics for calculating consistency. However, for expository purposes here, we use one that is intuitive and easy to interpret. First, we determined the annotation for each item (i.e., each verb/frame combination) by majority vote. We then considered how many verbs in each class had the same annotation in any given syntactic frame.

For example, suppose a class had 10 verbs and 2 frames. In the first frame, 8 verbs received the same annotation and 2 received others. The consistency for this class/frame combination is 80%. In the second frame, 6 verbs received the same annotation and 4 verbs received others. The consistency for this class/frame combination is 60%. The consistency for the class as a whole is the average across frames: 70%.

4.3.2 Results

Mean consistency averaged across classes is shown for each task in Table 2. As expected, consistency was lowest for EVALUATION, which is not expected to necessarily correlate with syntax. Interestingly, consistency for EVALUATION was nonetheless well above floor. This is perhaps not surprising: two sentences that have the same values for PHYSICAL CHANGE, APPLICATION OF FORCE, PHYSICAL CONTACT, CHANGE OF MENTAL STATE, MENTAL STATE, and LOCATION CHANGE are, on average, also likely to be both good or both bad.

Consistency was much higher for the other tasks, and in fact was close to ceiling for most of them. It remains to be seen whether the items that deviate from the mode represent true differences in semantics or reflect merely noise. One way of addressing this question is to collect additional annotations for those items that deviate from the mode.

4.4 Verb semantics

For each syntactic frame in each class, we determined the most common annotation. This is summarized in Table 3. The semantic annotation depended on syntactic frame nearly 1/4 of the time.⁴

These frequently matched VerbNet's semantics, though not always. For instance, annotators judged that class 18.1 verbs in the NP V NP PP.INSTRUMENT entailed movement on the part of the instrument (*Sally hit the ball with the stick*) – something not reflected in VerbNet.

5 Conclusion and Future Work

Results of Phase 1 provide support for the Semantic Consistency Hypothesis, at least as a strong bias. More work will be needed to determine the strength of that bias. The findings are largely consistent with VerbNet's semantics, but changes are indicated in some cases.

We find that inter-annotator agreement is sufficiently high that annotation can be done effectively using the modal response with an average of 6-7 responses per item. We are currently investigating whether we can achieve better reliability with fewer responses per item by taking into account an individual annotator's history across items, as recent work suggests is possible (Passonneau and Carpenter, 2013; Rzhetsky et al., 2009; Whitehill et al., 2009).

Thus, crowd-sourcing VerbNet semantic entailments appears to be both feasible and productive. Data-collection continues. Phase 2, which added over 10 new verb classes, is complete. Phase 3, which includes both new classes and new entailments, has been launched.

Acknowledgments

We gratefully acknowledge the support of the National Science Foundation Grant NSF-IIS-1116782, DARPA Machine Reading FA8750-09-C-0179, and funding from the Ruth L. Kirschstein National Research Service Award. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

⁴Note that this table was calculated based on whether the semantic feature was present or not. In many cases, the data was significantly richer. For instance, for APPLICATION OF FORCE, annotators determined which participant in the event was applying the force.

References

- Ben Ambridge, Julian Pine, Caroline Rowland, Franklin Chang, and Amy Bidgood. 2013. The retreat from overgeneralization in child language acquisition: word learning, morphology and verb argument structure. *Wiley Interdisciplinary Reviews: Cognitive Science*, 4(1):47–62.
- Libby Barak, Afsaneh Fazly, and Suzanne Stevenson. 2012. Modeling the acquisition of mental state verbs. In *Proceedings of the 3rd Workshop on Cognitive Modeling and Computational Linguistics*, pages 1–10. Association for Computational Linguistics.
- Leda Cosmides and John Tooby. 1992. Cognitive adaptations for social exchange. *The Adapted Mind*, pages 163–228.
- Edward Gibson and Evelina Fedorenko. 2013. The need for quantitative methods in syntax and semantics research. *Language and Cognitive Processes*, 28(1-2):88–124.
- Ana-Maria Giuglea and Alessandro Moschitti. 2006. Shallow semantic parsing based on framenet, verbnet and propbank. In *Proceedings of the 217th European Conference on Artificial Intelligence*, pages 563–567, Amsterdam, The Netherlands, The Netherlands. IOS Press.
- Lila Gleitman. 1990. The structural sources of verb meanings. *Language Acquisition*, 1(1):3–55.
- Adele E. Goldberg. 1995. *Constructions: A Construction Grammar approach to argument structure*. University of Chicago Press.
- Eric Joanis, Suzanne Stevenson, and David James. 2008. A general feature space for automatic verb classification. *Natural Language Engineering*, 14(3):337–367.
- Beth Levin and Malka Rappaport Hovav. 2005. *Argument Realization*. Cambridge University Press.
- Beth Levin. 1993. *English Verb Classes and Alternations: A preliminary Investigation*. University of Chicago press.
- Diana Maynard, Adam Funk, and Wim Peters. 2009. Using lexico-syntactic ontology design patterns for ontology creation and population. In *Proc. of the Workshop on Ontology Patterns*.
- Christopher Parisien and Suzanne Stevenson. 2011. Generalizing between form and meaning using learned verb classes. In *Proceedings of the 33rd Annual Meeting of the Cognitive Science Society*. Cite-seer.
- Rebecca J Passonneau and Bob Carpenter. 2013. The benefits of a model of annotation. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 187–195.
- Steven Pinker. 1984. *Language Learnability and Language Development*. Harvard University Press.
- Steven Pinker. 1989. *Learnability and Cognition: The Acquisition of Argument Structure*. MIT Press.
- Massimo Poesio, Jon Chamberlain, Udo Kruschwitz, Livio Robaldo, and Luca Ducceschi. 2012. The phrase detective multilingual corpus, release 0.1. In *Collaborative Resource Development and Delivery Workshop Programme*, page 34.
- Andrey Rzhetsky, Hagit Shatkay, and W John Wilbur. 2009. How to get the most out of your curation effort. *PLoS Computational Biology*, 5(5):113.
- Robert S Swier and Suzanne Stevenson. 2004. Un-supervised semantic role labeling. In *Proceedings of the Generative Lexicon Conference*, volume 95, page 102.
- Jacob Whitehill, Paul Ruvolo, Tingfan Wu, Jacob Bergsma, and Javier R Movellan. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems*, volume 22, pages 2035–2043.
- Annie Zaenen, Daniel G Bobrow, and Cleo Condoravdi. 2008. The encoding of lexical implications in verbnet: Predicates of change of locations. In *Language Resources Evaluation Conference*.

A Corpus of Sentence-level Revisions in Academic Writing: A Step towards Understanding Statement Strength in Communication

Chenhao Tan

Dept. of Computer Science
Cornell University
chenhao@cs.cornell.edu

Lillian Lee

Dept. of Computer Science
Cornell University
llee@cs.cornell.edu

Abstract

The strength with which a statement is made can have a significant impact on the audience. For example, international relations can be strained by how the media in one country describes an event in another; and papers can be rejected because they overstate or understate their findings. It is thus important to understand the effects of statement strength. A first step is to be able to distinguish between strong and weak statements. However, even this problem is understudied, partly due to a lack of data. Since strength is inherently relative, *revisions* of texts that make claims are a natural source of data on strength differences. In this paper, we introduce a corpus of sentence-level revisions from academic writing. We also describe insights gained from our annotation efforts for this task.

1 Introduction

It is important for authors and speakers to find the appropriate “pitch” to convey a desired message to the public. Indeed, sometimes heated debates can arise around the choice of statement strength. For instance, on March 1, 2014, an attack at Kunming’s railway station left 29 people dead and more than 140 others injured.¹ In the aftermath, Chinese media accused Western media of “soft-pedaling the attack and failing to state clearly that it was an act of *terrorism*”.² In particular, regarding the statement by the US embassy that referred to this incident as the “terrible and senseless act of violence in Kunming”, a Weibo user posted “If you say that the Kunming attack is a ‘terrible and

senseless act of violence’, then the 9/11 attack can be called a ‘regrettable traffic incident’”.³

This example is striking but not an isolated case, for settings in which one party is trying to convince another are pervasive; scenarios range from court trials to conference submissions. Since the strength and scope of an argument can be a crucial factor in its success, it is important to understand the effects of statement strength in communication.

A first step towards addressing this question is to be able to distinguish between strong and weak statements. As strength is inherently relative, it is natural to look at *revisions* that change statement strength, which we refer to as “*strength changes*”. Though careful and repeated revisions are presumably ubiquitous in politics, legal systems, and journalism, it is not clear how to collect them; on the other hand, revisions to research papers may be more accessible, and many researchers spend significant time on editing to convey the right message regarding the strength of a project’s contributions, novelty, and limitations. Indeed, statement strength in science communication matters to writers: understating contributions can affect whether people recognize the true importance of the work; at the same time, overclaiming can cause papers to be rejected.

With the increasing popularity of e-print services such as the arXiv⁴, strength changes in scientific papers are becoming more readily available. Since the arXiv started in 1991, it has become “the standard repository for new papers in mathematics, physics, statistics, computer science, biology, and other disciplines” (Krantz, 2007). An intriguing observation is that many researchers submit multiple versions of the same paper on arXiv. For instance, among the 70K papers submitted in

¹http://en.wikipedia.org/wiki/2014_Kunming_attack

²<http://sinosphere.blogs.nytimes.com/2014/03/03/u-n-security-council-condemns-terrorist-attack-in-kunming/>

³http://www.huffingtonpost.co.uk/2014/03/03/china-kunming-911_n_4888748.html

⁴<http://arxiv.org/>

ID	Pairs
1	S1: The algorithm is <i>studied</i> in this paper . S2: The algorithm is <i>proposed</i> in this paper .
2	S1: ... circadian pattern and burstiness in <i>human communication activity</i> . S2: ... circadian pattern and burstiness in <i>mobile phone communication</i> .
3	S1: ... using minhash techniques , <i>at a significantly lower cost and with same privacy guarantees</i> . S2: ... using minhash techniques , <i>with lower costs</i> .
4	S1: the rows and columns of the covariate matrix <i>then</i> have <i>certain physical</i> meanings ... S2: the rows and columns of the covariate matrix <i>could</i> have <i>different</i> meanings ...
5	S1: they maximize the expected revenue of the seller but <i>induce efficiency loss</i> . S2: they maximize the expected revenue of the seller but <i>are inefficient</i> .

Table 1: Examples of potential strength differences.

2011, almost 40% (27.7K) have multiple versions. Many differences between these versions constitute a source of valid and motivated strength differences, as can be seen from the sentential revisions in Table 1. Pair 1 makes the contribution seem more impressive by replacing “studied” with “proposed”. Pair 2 downgrades “human communication activity” to “mobile phone communication”. Pair 3 removes “significantly” and the emphasis on “same privacy guarantees”. Pair 4 shows an insertion of hedging, a relatively well-known type of strength reduction. Pair 5 is an interesting case that shows the complexity of this problem: on the one hand, S2 claims that something is “inefficient”, which is an absolute statement, compared to “efficiency loss” in S1, where the possibility of efficiency still exists; on the other hand, S1 employs an active tone that emphasizes a causal relationship.

The main contribution of this work is to provide the first large-scale corpus of sentence-level revisions for studying a broad range of variations in statement strength. We collected labels for a subset of these revisions. Given the possibility of all kinds of disagreement, the fair level of agreement (Fleiss’ Kappa) among our annotators was decent. But in some cases, the labels differed from our expectations, indicating that the general public can interpret the strength of scientific statements differently from researchers. The participants’ comments may further shed light on science communication and point to better ways to define and understand strength differences.

2 Related Work and Data

Hedging, which can lead to strength differences, has received some attention in the study of science

communication (Salager-Meyer, 2011; Lewin, 1998; Hyland, 1998; Myers, 1990). The CoNLL 2010 Shared Task was devoted to hedge detection (Farkas et al., 2010). Hedge detection was also used to understand scientific framing in debates over genetically-modified organisms in food (Choi et al., 2012).

Revisions on Wikipedia have been shown useful for various applications, including spelling correction (Zesch, 2012), sentence compression (Yamangil and Nelken, 2008), text simplification (Yatskar et al., 2010), paraphrasing (Max and Wisniewski, 2010), and textual entailment (Zanzotto and Pennacchiotti, 2010). But none of the categories of Wikipedia revisions previously examined (Daxenberger and Gurevych, 2013; Bronner and Monz, 2012; Mola-Velasco, 2011; Potthast et al., 2008; Daxenberger and Gurevych, 2012) relate to statement strength. After all, the objective of editing on Wikipedia is to present neutral and objective articles.

Public datasets of science communication are available, such as the ACL Anthology,⁵ collections of NIPS papers,⁶ and so on. These datasets are useful for understanding the progress of disciplines or the evolution of topics. But the lack of edit histories or revisions makes them not immediately suitable for studying strength differences. Recently, there have been experiments with *open peer review*.⁷ Records from open reviewing can provide additional insights into the revision process once enough data is collected.

⁵<http://aclweb.org/anthology/>

⁶<http://nips.djvuzone.org/txt.html>

⁷<http://openreview.net>

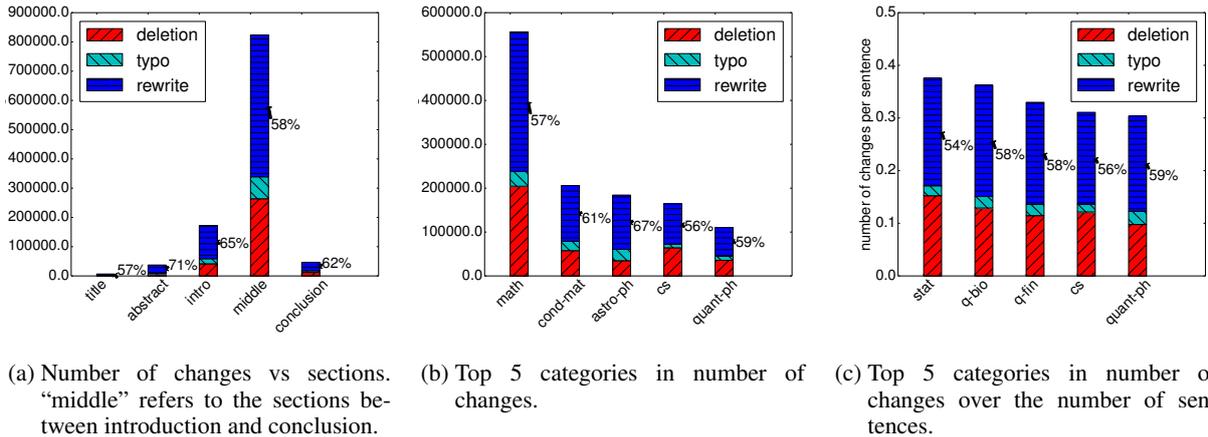


Figure 1: In all figures, different colors indicate different types of changes.

3 Dataset Description

Our main dataset was constructed from all papers submitted in 2011 on the arXiv. We first extracted the textual content from papers that have multiple versions of tex source files. All mathematical environments were ignored. Section titles were not included in the final texts but are used in alignment.

In order to align the first version and the final version of the same paper, we first did macro alignment of paper sections based on section titles. Then, for micro alignment of sentences, we employed a dynamic programming algorithm similar to that of Barzilay and Elhadad (2003). Instead of cosine similarity, we used an idf-weighted longest-common-subsequence algorithm to define the similarity between two sentences, because changes in word ordering can also be interesting. Formally, the similarity score between sentence i and sentence j is defined as

$$Sim(i, j) = \frac{\text{Weighted-LCS}(S_i, S_j)}{\max(\sum_{w \in S_i} idf(w), \sum_{w \in S_j} idf(w))},$$

where S_i and S_j refer to sentence i and sentence j . Since it is likely that a new version adds or deletes a large sequence of sentences, we did not impose a skip penalty. We set the mismatch penalty to 0.1.⁸

In the end, there are 23K papers where the first version was different from the last version.⁹ We

⁸We did not allow cross matching (i.e., $i \rightarrow j-1, i-1 \rightarrow j$), since we thought matching this case as $(i-1, i) \rightarrow j$ or $i \rightarrow (j, j-1)$ can provide context for annotation purposes. But in the end, we focused on labeling very similar pairs. This decision had little effect.

⁹This differs from the number in Section 1 because articles may not have the tex source available, or the differences between versions may be in non-textual content.

categorize sentential revisions into the following three types:

- Deletion: we cannot find a match in the final version.
- Typo: all sequences in a pair of matched sentences are typos, where a sequence-level typo is one where the edit distance between the matched sequences is less than three.
- Rewrite: matched sentences that are not typos. This type is the focus of this study.

What kinds of changes are being made? One might initially think that typo fixes represent a large proportion of revisions, but this is not correct, as shown in Figure 1a. Deletions represent a substantial fraction, especially in the middle section of a paper. But it is clear that the majority of changes are rewrites; thus revisions on the arXiv indeed provide a great source for potential strength differences.

Who makes changes? Figure 1b shows that the Math subarchive makes the largest number of changes. This is consistent with the mathematics community’s custom of using the arXiv to get findings out early. In terms of changes per sentence (Figure 1c), statistics and quantitative studies are the top subareas.

Further, Figure 2 shows the effect of the number of authors. It is interesting that both in terms of sheer number and percentage, single-authored papers have the most changes. This could be because a single author enjoys greater freedom and has stronger motivation to make changes, or because multiple authors tend to submit a more polished initial version. This echoes the finding in Posner

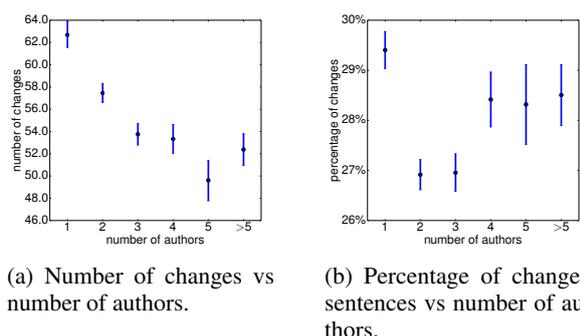
You should mark S2 as **Stronger** if

- (R1) S2 strengthens the degree of some aspect of S1, for example, S1 has the word "better", whereas S2 uses "best", or S2 removes the word "possibly"
- (R2) S2 adds more evidence or justification (we *don't* count adding details)
- (R3) S2 sounds more impressive in some other way: the authors' work is more important/novel-/elegant/applicable/etc.

If instead S1 is stronger than S2 according to the reasons above, select **Weaker**. If the changes aren't strengthenings or weakenings according to the reason above, select No Strength Change.

If there are both strengthenings and weakenings, or you find that it is really hard to tell whether the change is stronger or weaker, then select I can't tell.

Table 2: Definition of labels in our labeling tasks.



(a) Number of changes vs number of authors.

(b) Percentage of changed sentences vs number of authors.

Figure 2: Error bars represent standard error. (a): up until 5 authors, a larger number of authors indicates a smaller number of changes. (b): percentage is measured over the number of sentences in the first version; there is an interior minimum where 2 or 3 authors make the smallest percentage of sentence changes on a paper.

and Baecker (1992) that the collaborative writing process differs considerably from individual writing. Also, more than 25% of the first versions are changed, which again shows that substantive edits are being made in these resubmissions.

4 Annotating Strength Differences

In order to study statement strength, reliable strength-difference labels are needed. In this section, we describe how we tried to define strength differences, compiled labeling instructions, and gathered labels using Amazon Mechanical Turk.

Label definition and collection procedure. We focused on matched sentences from abstracts and introductions to maximize the proportion of strength differences (as opposed to factual/no strength changes). We required pairs to have similarity score larger than 0.5 in our labeling task to make pairs more comparable. We also replaced

all math environments with "[MATH]".¹⁰ We obtained 108K pairs that satisfy the above conditions, available at <http://chenhaot.com/pages/statement-strength.html>. To create the pool of pairs for labeling, we randomly sampled 1000 pairs and then removed pairs that we thought were processing errors.

We used Amazon Mechanical Turk. It may initially seem surprising to have annotations of technical statements not done by domain experts; we did this intentionally because it is common to communicate unfamiliar topics to the public in political and science communication (we comment on non-expert rationales later). We use the following set of labels: *Stronger*, *Weaker*, *No Strength Change*, *I can't tell*. Table 2 gives our definitions. The instructions included 8 pairs as examples and 10 pairs to label as a training exercise. Participants were then asked to choose labels and write mandatory comments for 50 pairs. According to the comments written by participants, we believe that they did the labeling in good faith.

Quantitative overview. We collected 9 labels each for 500 pairs. Among the 500 pairs, Fleiss' Kappa was 0.242, which indicates fair agreement (Landis and Koch, 1977). We took a conservative approach and only considered pairs with an absolute majority label, i.e., at least 5 of 9 labelers chose the same label. There are 386 pairs that satisfy this requirement (93 weaker, 194 stronger, 99 no change). On this subset of pairs, Fleiss' Kappa is 0.322, and 74.4% of pairs were strength changes. Considering all the possible disagreement, this result was acceptable.

Qualitative observations. We were excited about the labels from these participants: despite

¹⁰These decisions were made based on the results and feedback that we got from graduate students in an initial labeling.

ID	Matched sentences and comments
1	S1: ... using data from numerics and experiments . S2: ... using data sets from numerics in the point particle limit and one experimental data set . (stronger) S2 is more specific in its description which seems stronger. (weaker) "one experimental data set" weakens the sentence
2	S1: we also proved that if [MATH] is sufficiently homogeneous then ... S2: we also proved that if [MATH] is not totally disconnected and sufficiently homogeneous then ... (stronger) We have more detail/proof in S2 (stronger) the words "not totally disconnected" made the sentence sound more impressive.
3	S1: we also show in general that vectors of products of jack vertex operators form a basis of symmetric functions . S2: we also show in general that the images of products of jack vertex operators form a basis of symmetric functions . (weaker) Vectors sounds more impressive than images (weaker) sentence one is more specific
4	S1: in the current paper we discover several variants of qd algorithms for quasiseparable matrices . S2: in the current paper we adapt several variants of qd algorithms to quasiseparable matrices . (stronger) in S2 Adapt is stronger than just the word discover. adapt implies more of a proactive measure. (stronger) s2 sounds as if they're doing something with specifics already, rather than hunting for a way to do it

Table 3: Representative examples of surprising labels, together with selected labeler comments.

the apparent difficulty of the task, we found that many labels for the 386 pairs were reasonable. However, in some cases, the labels were counter-intuitive. Table 3 shows some representative examples.

First, participants tend to take details as evidence even when these details are not germane to the statement. For pair 1, while one turker pointed out the decline in number of experiments, most turkers simply labeled it as stronger because it was more specific. "Specific" turned out to be a common reason used in the comments, even though we said in the instructions that only additional justification and evidence matter. This echoes the finding in Bell and Loftus (1989) that even unrelated details influenced judgments of guilt.

Second, participants interpret constraints/conditions not in strictly logical ways, seeming to care little about scope at times. For instance, the majority labeled pair 2 as "stronger". But in S2 for that pair, the result holds for strictly fewer possible worlds. But it should be said that there are cases that labelers interpreted logically, e.g., "compelling evidence" subsumes "compelling experimental evidence".

Both of the above cases share the property that they seem to be correlated with a tendency to judge lengthier statements as stronger. Another interesting case that does not share this characteristic is that participants can have a different understanding of domain-specific terms. For pair 3, the majority thought that "vectors" sounds more impressive than "images"; for pair 4, the majority considered "adapt" stronger than "discover". This issue is common when communicating new topics to the public not only in science commu-

nication but also in politics and other scenarios. It may partly explain miscommunications and misinterpretations of scientific studies in journalism.¹¹

5 Looking ahead

Our observations regarding the annotation results raise questions regarding what is a generalizable way to define strength differences, how to use the labels that we collected, and how to collect labels in the future. We believe that this corpus of sentence-level revisions, together with the labels and comments from participants, can provide insights into better ways to approach this problem and help further understand strength of statements.

One interesting direction that this enables is a potentially new kind of learning problem. The comments indicate features that humans think salient. Is it possible to automatically learn new features from the comments?

The ultimate goal of our study is to understand the effects of statement strength on the public, which can lead to various applications in public communication.

Acknowledgments

We thank J. Baldridge, J. Boyd-Graber, C. Callison-Burch, and the reviewers for helpful comments; P. Ginsparg for providing data; and S. Chen, E. Kozyri, M. Lee, I. Lenz, M. Ott, J. Park, K. Raman, M. Reitblatt, S. Roy, A. Sharma, R. Sipos, A. Swaminathan, L. Wang, W. Xie, B. Yang and the anonymous annotators for all their labeling help. This work was supported in part by NSF grant IIS-0910664 and a Google Research Grant.

¹¹<http://www.phdcomics.com/comics/archive.php?comicid=1174>

References

- Regina Barzilay and Noemie Elhadad. 2003. Sentence alignment for monolingual comparable corpora. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 25–32.
- Brad E Bell and Elizabeth F Loftus. 1989. Trivial persuasion in the courtroom: The power of (a few) minor details. *Journal of Personality and Social Psychology*, 56(5):669.
- Amit Bronner and Christof Monz. 2012. User Edits Classification Using Document Revision Histories. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 356–366.
- Eunsol Choi, Chenhao Tan, Lillian Lee, Cristian Danescu-Niculescu-Mizil, and Jennifer Spindel. 2012. Hedge detection as a lens on framing in the GMO debates: A position paper. In *Proceedings of the Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics*, pages 70–79.
- Johannes Daxenberger and Iryna Gurevych. 2012. A Corpus-Based Study of Edit Categories in Featured and Non-Featured Wikipedia Articles. In *COLING*, pages 711–726.
- Johannes Daxenberger and Iryna Gurevych. 2013. Automatically Classifying Edit Categories in Wikipedia Revisions. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*.
- Richárd Farkas, Veronika Vincze, György Móra, János Csirik, and György Szarvas. 2010. The CoNLL-2010 shared task: Learning to detect hedges and their scope in natural language text. In *CoNLL—Shared Task*, pages 1–12.
- Ken Hyland. 1998. *Hedging in scientific research articles*. John Benjamins Pub. Co., Amsterdam; Philadelphia.
- Steven G. Krantz. 2007. How to Write Your First Paper. *Notices of the AMS*.
- J. Richard Landis and Gary G. Koch. 1977. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174.
- Beverly A. Lewin. 1998. Hedging: Form and function in scientific research texts. In *Genre Studies in English for Academic Purposes*, volume 9, pages 89–108. Universitat Jaume I.
- Aurlien Max and Guillaume Wisniewski. 2010. Mining Naturally-occurring Corrections and Paraphrases from Wikipedia’s Revision History. In *Proceedings of The seventh international conference on Language Resources and Evaluation*.
- Santiago M Mola-Velasco. 2011. Wikipedia Vandalism Detection. In *Proceedings of the 20th International Conference Companion on World Wide Web*, pages 391–396.
- Greg Myers. 1990. *Writing biology: Texts in the social construction of scientific knowledge*. University of Wisconsin Press, Madison, Wis.
- Iлона R Posner and Ronald M Baecker. 1992. How people write together [groupware]. In *System Sciences, 1992. Proceedings of the Twenty-Fifth Hawaii International Conference on*, pages 127–138.
- Martin Potthast, Benno Stein, and Robert Gerling. 2008. Automatic Vandalism Detection in Wikipedia. In *Advances in Information Retrieval*, pages 663–668. Springer Berlin Heidelberg.
- Françoise Salager-Meyer. 2011. Scientific discourse and contrastive linguistics: hedging. *European Science Editing*, 37(2):35–37.
- Elif Yamangil and Rani Nelken. 2008. Mining Wikipedia Revision Histories for Improving Sentence Compression. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 137–140.
- Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from Wikipedia. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 365–368.
- Fabio Massimo Zanzotto and Marco Pennacchiotti. 2010. Expanding textual entailment corpora from Wikipedia using co-training. In *Proceedings of the 2nd Workshop on Collaboratively Constructed Semantic Resources*.
- Torsten Zesch. 2012. Measuring Contextual Fitness Using Error Contexts Extracted from the Wikipedia Revision History. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 529–538.

Determiner-Established Deixis to Communicative Artifacts in Pedagogical Text

Shomir Wilson^{1,2} and Jon Oberlander¹

¹School of Informatics, University of Edinburgh, United Kingdom

²School of Computer Science, Carnegie Mellon University, USA

shomir@cs.cmu.edu, jon@inf.ed.ac.uk

Abstract

Pedagogical materials frequently contain deixis to communicative artifacts such as textual structures (e.g., sections and lists), discourse entities, and illustrations. By relating such artifacts to the prose, deixis plays an essential role in structuring the flow of information in informative writing. However, existing language technologies have largely overlooked this mechanism. We examine properties of deixis to communicative artifacts using a corpus rich in determiner-established instances of the phenomenon (e.g., “this section”, “these equations”, “those reasons”) from Wikibooks, a collection of learning texts. We use this corpus in combination with WordNet to determine a set of word senses that are characteristic of the phenomenon, showing its diversity and validating intuitions about its qualities. The results motivate further research to extract the connections encoded by such deixis, with the goals of enhancing tools to present pedagogical e-texts to readers and, more broadly, improving language technologies that rely on deictic phenomena.

1 Introduction

Deixis often appears in written language as an anaphoric mechanism to refer to communicative entities in a document. Such deixis can have a variety of referent types. For example, consider *that idea* in Sentence (1), *those names* in (2), *this section* in (3), and *these figures* in (4):

- (1) That idea has been challenged by many.
- (2) Those names are Welsh in origin.
- (3) In this section, we cover some early work.
- (4) Quantities in these figures are approximate.

The kinds of deixis represented in (1) and (2) are similar to discourse deixis (Webber, 1991) and textual deixis (Lyons, 1977), respectively. Sentence (3) contains deixis to a structural element of a document (Paraboni and Deemter,

2006), and (4) contains an example of deixis to illustrative items such as figures or examples. We collectively term such deictic acts as *communicative deixis* (CD for brevity), recognizing their shared characteristics, and we name their referents *communicative artifacts* (CAs). Prior studies have focused on narrow varieties of CD (such as those identified above), leaving unknown their properties when viewed together as a whole. Moreover, efforts to automatically identify or resolve CD have been piecemeal at best. Given the complexity of the referents, conventional tools for coreference or anaphora resolution are poorly applicable.

This paper describes analysis of the first collection of instances of deixis in English targeted to refer to a broad variety of CAs. Texts from the website *Wikibooks* are used, for the intuitive density of CD in pedagogical material and the potential value of augmenting them with interpretive metadata. The diversity of referents in this corpus enables new inferences on the composition and relative frequencies of CD varieties in text. We focus on *determiner-established* instances, i.e., anaphoric noun phrases that begin with determiners *this*, *that*, *these*, or *those* (e.g., (1)-(4)). This focus has the advantage of collecting instances that explicitly identify the relevant capacities of their referents (e.g., (1) reifies its referent as an “idea”).

The remainder of this paper is structured as follows. Section 2 surveys related work on deixis to specific types of CAs. Section 3 describes the text source for this study and the procedure used to collect and label instances. Section 4 describes our use of WordNet to characterize CAs, resulting in an ontology of such referents and inter-annotator agreement results for labeling of artifact types. Finally, Section 5 provides some conclusions and directions for future work.

2 Related Work

The value of CD in pedagogical contexts has been established by studies such as those by Mayer (2009) and Buisine and Martin (2007). Those motivate our work to fill the present lack of corpus-based linguistic knowledge of the phenomenon. Also, although spatial deixis falls beyond the scope of this paper, we acknowledge the efforts of others such as Gergle et al. (2013) to study its value in collaborative communication.

Prior works have examined *discourse deixis* in text, though little attention has been given to CD as a phenomenon or deixis to other CAs. Seminal papers by Webber (1988, 1991) established the importance of discourse deixis, although they focused upon demonstrative pronouns such as “this” or “that”. Many efforts have addressed discourse deixis in the context of anaphora; these include Poesio and Artstein’s (2008), who created a corpus of anaphoric relations inclusive of (but not limited to) discourse. Their collection included 455 instances of discourse deixis, although they noted ambiguity in the set of markables. Dipper and Zinsmeister (2012) also addressed discourse deixis through anaphora resolution and produced a collection of 225 abstract anaphors out of 643 candidate instances.

Prior studies of *shell nouns* revealed capacities of referents similar to a subset of those found in our work. Such nouns are used anaphorically to refer to complex, proposition-like pieces of information such as points, assumptions, or acts (Schmid, 2000). Kolhatkar et al. (2013) noted the pervasiveness of shell nouns in text and their tendency to “characterize and label” their antecedents. However, such antecedents only partly intersect with CAs. The set of shell nouns studied by Schmid did not include typical document entities such as *section*, *figure*, or *list*. Simultaneously, the set included many nouns with little or no relevance as CAs, such as *fury*, *miracle*, and *pride*.

The task of identifying CD in text and referent CAs bears some similarity to coreference resolution. However, coreference resolvers tried by the authors (namely CoreNLP (Recasens et al., 2013), ArkRef (O’Connor and Heilman, 2013) and the work of Roth and Bengston (2008)) were ineffective at this task. We posit that many CAs are not noun phrases, which makes them difficult or inappropriate to characterize as referring expressions. This limits the effectiveness of traditional approaches to coreference resolution toward the present problem.

Statistic	Total	Min.	Median	Mean	Max.
Words	2883178	1721	20337	23633	57465
Sentences	114474	71	832	938	2121
Candidates	10495	4	85	86	285

Table 1. Statistics for the 122 selected printable Wikibooks and the candidate instances of CD.

Our results are further distinct from prior work by focusing on the communicative capacities of a variety of referents represented in documents. However, the present focus upon determiner-established phrases is more exclusive, and our results do not include demarcation of referents. We posit that the tradeoff is worthwhile, given limited prior work on identifying CD and the lack of prior efforts to study CAs other than discourse entities.

3 Corpus Creation

Textbooks from *Wikibooks* were chosen to supply pedagogical text. Among the alternatives, this source provided the largest volume of material with a license amenable to corpus redistribution. Moreover, the collection of English language textbooks on the site covers a diverse set of topics and contains samples from a variety of writers. Below we describe our text pre-processing and then explain how candidate instances of CD were identified.

3.1 Source Material

To simplify collection and processing, 122 Wikibooks textbooks with printable versions were selected for use. Contained in this set are textbooks in eleven different subject areas, such as computing, humanities, and the sciences. In preparation for analysis, the documents were POS tagged and parsed by the Stanford CoreNLP suite (Socher et al., 2013; Toutanova et al., 2003). Table 1 presents some statistics on the texts in aggregate. They illustrate the substantial size of most texts, though a few were freshly started or incomplete. Overall, the corpus is comparable in size with corpora from efforts cited in Section 2, though text genera and sought markables vary.

Next, potential instances of CD were identified. Such instances were noun phrases beginning with determiners *this*, *that*, *these*, or *those*. We include *these* and *those* to collect CD to sets of entities, a nuance absent from any previous work. 9252 sentences, or 8% of the corpus, contained at least one potential instance.

Lemma	Freq.	Lemma	Freq.
page	314	function	83
book	287	chapter	73
case	249	information	70
example	126	problem	69
point	121	value	62
section	116	type	59
way	112	process	56
option	102	feature	56
time	101	number	54
message	93	text	54

Table 2. The 20 most frequent head nouns in candidate instances.

This collection contained substantial boilerplate text, and sentences that appeared verbatim in at least ten different books were discarded. This filtering produced a set of 7613 *candidate instances*. Table 2 shows the most frequent head nouns in candidate instances. Some resemble the shell nouns of prior work, but the presence of others illustrates the diversity of CD. Diversity was expected from pedagogical texts and validates Wikibooks as a rich source of CD.

We conducted a preliminary survey of the corpus contents by reading a random selection of 10% of candidates and judging their statuses as instances of CA. Table 3 shows examples of candidate instances, categorized by the foci of prior studies (cited in the Introduction) of CD phenomena. The researchers estimated that 48% of candidates were instances of CD, although directly labeling large numbers of candidates was deemed impractical. Instead, we noted that the word sense of the noun in a candidate instance is an important (albeit not definitive) indication of its CD status. Accordingly, we shift our focus from individual candidate instances to words that appear in them (i.e., lemmas) and word senses.

For each synset gloss, perform the following:

Imagine instantiating the type represented by the gloss. Judge its suitability for the following statements.

(1) [an instantiation of the type] is about a topic.

(2) [an instantiation of the type] is intended to communicate an idea.

(3) [an instantiation of the type] can be produced in a document or as a document to convey information.

If at least two of the three statements above are coherent, mark 'y' for the gloss. Otherwise, mark 'n'.

Figure 1. Instructions given to annotators.

3.2 Word Senses

The noun in an instance of CD has a doubly salient role in CA, by providing a cue to the intended referent and also by reifying the referent. For example, an illustrating referent might be referred to as “this example” or “this ideal”, with divergent consequences. The noun choice semantically identifies the relevant capacity of the referent, affecting its message.

To identify the varieties and characteristics of CD in pedagogical text, we examine in aggregate the senses of those words that appear in candidate phrases in the corpus. WordNet 3.0 (Fellbaum, 1998) was chosen to provide an ontological structure for relevant word senses and thus for CAs. First, synsets for the 27 most frequent nouns in candidate phrases were collected, irrespective of viability for CD. This covered 34% of candidate instances and resulted in a set of 200 synsets. Their glosses were labeled as viable or non-viable for CD by two expert annotators, who first worked separately and then collaborated to resolve differences in their annotations.

Category	Examples
Structural	Many of the resources listed elsewhere in this section have...
	In this chapter , we will show you how to draw...
Illustrative	Consider these sentences : [followed by example sentences]
	[following a source code fragment] ...the first time the computer sees this statement , ‘a’ is zero, so it is less than 10.
Discourse	Utilizing this idea , subunit analogies were invented...
	In this case , you’ve narrowed the topic down to “Badges.”
Non-CD	Devices similar to resistors turn this energy into light, motion...
	What type of things does a person in that career field know?

Table 3. Examples of candidate instances. Bold text denotes the determiner and head noun in each instance. Sentences are truncated in the table for brevity.

Figure 1 shows the annotation instructions, which were designed to address the combined range of CAs from prior work. To illustrate its application, consider the noun *chapter*. One gloss of *chapter* is “a subdivision of a written work; usually numbered and titled”. This sense clearly satisfies the third numbered statement in Figure 1. Coherency arguments for the first and second statements are less definitional, but both annotators decided at least one was satisfactory, leading to a *y* mark. Another gloss of *chapter* is “any distinct period in history or in a person’s life”. This sense fails to satisfy the second or third statement, leading to an *n* mark.

4 Results and Discussion

Resolving differences between the annotators’ labels produced a set of 62 synsets whose glosses characterized CAs. We refer to the sets of 200 synsets and 62 synsets as the CCS (candidates for communicative senses) and VCS (verified communicative senses) sets, respectively. We offer the complete results of our annotations online¹ to encourage further research on this topic. In this section we present inter-annotator agreement statistics and describe the composition of the VCS set using the structure of WordNet.

4.1 Inter-Annotator Agreement

The kappa statistic for category agreement between the two annotators was 0.70, with matching annotations on 174 of 200 senses. Although this metric is an imperfect indicator, this value is generally regarded as substantial (Viera and Garrett, 2005) albeit with some tentativeness (Carletta, 1996). The annotators respectively placed 33% and 30% of instances in the VCS set, suggesting general agreement on the distribution of labels irrespective of specific instances. The annotators agreed that some cases were difficult to label without context, and a combination of sense labeling and in-text instance labeling may be fruitful for future work.

4.2 Representation in WordNet

We use the structure of WordNet to illustrate the properties of CAs that VCS senses represent. To do this, the hypernym closure (i.e., the sequence(s) of hypernyms from a given synset to the root synset) was computed for each VCS sense. These “traces” were aggregated into a

Synset	CCS	VCS	Chg.
0 entity.n.01	217 / 217	72 / 72	0
1 abstraction.n.06	166 / 217	65 / 72	.14
2 psych._feature.n.01	51 / 166	15 / 65	-.08
2 communication.n.02	47 / 166	37 / 65	.29
2 attribute.n.02	24 / 166	2 / 65	-.11
2 group.n.01	18 / 166	4 / 65	-.05
2 measure.n.02	15 / 166	3 / 65	-.04
2 relation.n.01	11 / 166	4 / 65	.00
1 physical_entity.n.01	51 / 217	7 / 72	-.14
2 object.n.01	38 / 51	6 / 7	.11
2 causal_agent.n.01	7 / 51	0 / 7	-.14
2 thing.n.12	4 / 51	0 / 7	-.08
2 process.n.06	1 / 51	0 / 7	-.02
2 matter.n.03	1 / 51	1 / 7	.12

Table 4. Distributions of traces through the first two hyponym relations emanating from the root synset *entity.n.01*, for CCS and VCS. Fractions indicate the constituent weight of each synset.

reproduction of a subset of WordNet’s synsets and relations, resulting in a *de facto* ontology of CAs. The same procedure was performed for the CCS set to create an illustrative baseline.

Table 4 shows the structure of the most general synsets in the ontologies constructed from VCS and CCS traces. Fractions illustrate the relative constituent weight of each synset, by virtue of the traces that include it. For example, 65 of the 72 traces for VCS synsets pass through *abstraction.n.06*, and 37 of those 65 traces pass through *communication.n.02*. The total quantities of traces for CCS and VCS are greater than their respective set sizes because of a small number of synsets in those sets with multiple hyponym paths to the root. The rightmost column of Table 4 shows the decimal result of subtracting the CCS constituent weight fraction from the VCS fraction. Positive numbers indicate that the manual labeling of senses magnified the weight of a synset over the CCS baseline.

The constituent weights confirm some intuitions but also hold a few surprises. The vast majority of CAs are abstractions rather than physical entities, and most of the abstractions are “something that is communicated by or to or between people or groups” (the gloss of *communication.n.02*). Psychological features are also a substantial constituency, with traces to VCS synsets that represent words such as *method*, *plan*, and *question*. Most of the few VCS physical entities are communicative artifacts in their complete form (e.g., a book or a periodical issue). *Matter* as a physical entity may seem out of place in Table 4. The VCS synset responsible for its inclusion is *page.n.01*, which

¹ http://www.cs.cmu.edu/~shomir/wb_cd_study/

has the gloss “one side of one leaf (of a book or magazine or newspaper or letter etc.) or the written or pictorial matter it contains.” Both annotators believed it merited inclusion in VCS.

Finally, we observed that many VCS senses (58%) were not the first sense for their words, indicating different senses appear more often². This likely hinders word sense disambiguation of nouns in CD instances: the common baseline of first sense tagging is futile in these cases, and their extra-topical nature means that appropriate CA senses are not implied by the surrounding words (Wilson, 2011). This suggests that identification of CD instances may require a dedicated approach to word sense tagging.

5 Conclusion

The results of this study illustrate the significance of CD, both for the processing of pedagogical texts and for the broader project of understanding anaphora. Its pervasiveness and its diversity show its potential as a conduit for language technologies to enrich documents with pragmatic metadata. Our next effort will be to identify the referents of CD instances using knowledge from the present study of the character and distribution of those referents. CAs are represented by spans of content in a document (e.g., text or figures), and accordingly the identification of a CD referent will involve the selection of the correct span of content. We expect that the word sense of the noun in a CD phrase will limit the set of potentially relevant CAs, and that both localized features (such as paragraph position of a CD instance and the expected CA count) and document-level features (e.g., proximity of potential referents) will be valuable.

Acknowledgment

This research was supported by grant #1159236 from the US National Science Foundation’s International Research Fellowship Program.

References

- Bengtson, E. and Roth, D. (2008). Understanding the value of features for coreference resolution. In *Proc. EMNLP*.
- Buisine, S. and Martin, J.-C. (2007). The effects of speech–gesture cooperation in animated agents’ behavior in multimedia presentations. *Interacting with Computers*, 19(4), 484–493. doi:10.1016/j.intcom.2007.04.002
- Carletta, J. (1996). Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2), 249–254.
- Dipper, S. and Zinsmeister, H. (2012). Annotating abstract anaphora. In *Proc. LREC*, 46(1), 37–52. doi:10.1007/s10579-011-9160-1
- Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. Cambridge: MIT Press.
- Gergle, D., Kraut, R. E., and Fussell, S. R. (2013). Using visual information for grounding and awareness in collaborative tasks. *Human-Computer Interaction*, 28(1), 1–39.
- Kolhatkar, V., Zinsmeister, H., and Hirst, G. (2013). Interpreting anaphoric shell nouns using antecedents of cataphoric shell nouns as training data. In *Proc. EMNLP* (pp. 300–310).
- Lyons, J. (1977). *Semantics*. Cambridge University Press.
- Mayer, R. E. (2009). *Multimedia Learning*. Cambridge University Press.
- O’Connor, B. and Heilman, M. (2013). ARKref: A rule-based coreference resolution system. arXiv:1310.1975,
- Paraboni, I. and Deemter, K. (2006). Referring via document parts. In A. Gelbukh (Ed.), *Computational Linguistics and Intelligent Text Processing* (Vol. 3878, pp. 299–310). Springer Berlin Heidelberg. Retrieved from http://dx.doi.org/10.1007/11671299_31
- Poesio, M. and Artstein, R. (2008). Anaphoric annotation in the ARRAU Corpus. In *Proc. LREC*. Marrakech, Morocco: European Language Resources Association (ELRA).
- Recasens, M., Catherine de Marneffe, M., and Potts, C. (2013). The life and death of discourse entities: Identifying singleton mentions. In *Proc. NAACL*.
- Schmid, H.-J. (2000). *English Abstract Nouns as Conceptual Shells: From Corpus to Cognition*. Walter de Gruyter.

² The WordNet manual advises that senses are “generally” ordered by frequency.

- Socher, R., Bauer, J., Manning, C. D., and Ng, A. Y. (2013). Parsing with compositional vector grammars. In *Proc. ACL* (pp. 455–465).
- Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. NAACL*. doi:10.3115/1073445.1073478
- Viera, A. J., and Garrett, J. M. (2005). Understanding interobserver agreement: The kappa statistic. *Family Medicine*, 37(5), 360–363.
- Webber, B. L. (1988). Discourse deixis: Reference to discourse segments. In *Proc. ACL* (pp. 113–122). doi:10.3115/982023.982037
- Webber, B. L. (1991). Structure and ostension in the interpretation of discourse deixis. In *Natural Language and Cognitive Processes*.
- Wilson, S. (2011). *A Computational Theory of the Use-Mention Distinction in Natural Language*. University of Maryland at College Park. PhD Thesis, College Park, MD, USA.

Modeling Factuality Judgments in Social Media Text

Sandeep Soni

Tanushree Mitra

Eric Gilbert

Jacob Eisenstein

School of Interactive Computing
Georgia Institute of Technology

soni.sandeepb@gmail.com, {tmitra3,gilbert,jeisenst}@cc.gatech.edu

Abstract

How do journalists mark quoted content as certain or uncertain, and how do readers interpret these signals? Predicates such as *thinks*, *claims*, and *admits* offer a range of options for framing quoted content according to the author’s own perceptions of its credibility. We gather a new dataset of direct and indirect quotes from Twitter, and obtain annotations of the perceived certainty of the quoted statements. We then compare the ability of linguistic and extra-linguistic features to predict readers’ assessment of the certainty of quoted content. We see that readers are indeed influenced by such framing devices — and we find no evidence that they consider other factors, such as the source, journalist, or the content itself. In addition, we examine the impact of specific framing devices on perceptions of credibility.

1 Introduction

Contemporary journalism is increasingly conducted through social media services like Twitter (Lotan et al., 2011; Hermida et al., 2012). As events unfold, journalists and political commentators use quotes — often indirect — to convey potentially uncertain information and claims from their sources and informants, e.g.,



Figure 1: Indirect quotations in Twitter

A key pragmatic goal of such messages is to convey the provenance and uncertainty of the

quoted content. In some cases, the author may also introduce their own perspective (Lin et al., 2006) through the use of framing (Greene and Resnik, 2009). For instance, consider the use of the word *claims* in Figure 1, which conveys the author’s doubt about the indirectly quoted content.

Detecting and reasoning about the certainty of propositional content has been identified as a key task for information extraction, and is now supported by the FactBank corpus of annotations for newstext (Saurí and Pustejovsky, 2009). However, less is known about this phenomenon in social media — a domain whose endemic uncertainty makes proper treatment of factuality even more crucial (Morris et al., 2012). Successful automation of factuality judgments could help to detect online rumors (Qazvinian et al., 2011), and might enable new applications, such as the computation of reliability ratings for ongoing stories.

This paper investigates how linguistic resources and extra-linguistic factors affect perceptions of the certainty of quoted information in Twitter. We present a new dataset of Twitter messages that use FactBank predicates (e.g., *claim*, *say*, *insist*) to scope the claims of named entity sources. This dataset was annotated by Mechanical Turk workers who gave ratings for the factuality of the scoped claims in each Twitter message. This enables us to build a predictive model of the factuality annotations, with the goal of determining the full set of relevant factors, including the predicate, the source, the journalist, and the content of the claim itself. However, we find that these extra-linguistic factors do not predict readers’ factuality judgments, suggesting that the journalist’s own framing plays a decisive role in the credibility of the information being conveyed. We explore the specific linguistic feature that affect factuality judgments, and compare our findings with previously-proposed groupings of factuality-related predicates.

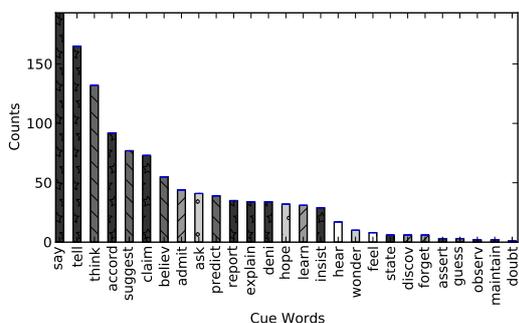


Figure 2: Count of cue words in our dataset. Each word is patterned according to its group, as shown in Figure 3.

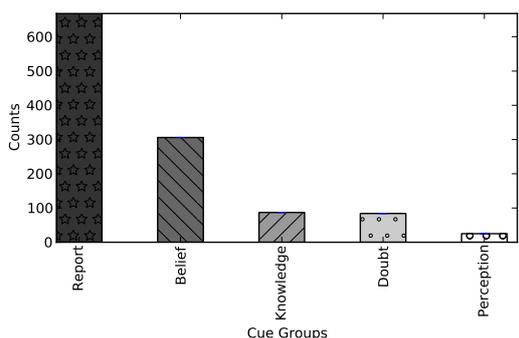


Figure 3: Count of cue groups in our dataset

2 Text data

We gathered a dataset of Twitter messages from 103 professional journalists and bloggers who work in the field of American Politics.¹ Tweets were gathered using Twitter’s streaming API, extracting the complete permissible timeline up to February 23, 2014. A total of 959,754 tweets were gathered, and most were written in early 2014.

Our interest in this text is specifically in quoted content — including “indirect” quotes, which may include paraphrased quotations, as in the examples in Figure 1. While labeled datasets for such quotes have been created (O’Keefe et al., 2012; Pareti, 2012), these are not freely available at present. In any case, the relevance of these datasets to Twitter text is currently unproven. Therefore, rather than train a supervised model to detect quotations, we apply a simple dependency-based heuristic.

- We focus on tweets that contain any member of a list of source-introducing predicates (we borrow the terminology of Pareti (2012) and call this the CUE). Our complete list — shown in Table 1 — was selected mainly from the examples presented by Saurí and Pustejovsky (2012),

¹We used the website <http://muckrack.com>.

Report	<i>say, report, tell, told, observe, state, accord, insist, assert, claim, maintain, explain, deny</i>
Knowledge	<i>learn, admit, discover, forget, forgot</i>
Belief	<i>think, thought, predict, suggest, guess, believe</i>
Doubt	<i>doubt, wonder, ask, hope</i>
Perception	<i>sense, hear, feel</i>

Table 1: Lemmas of source-introducing predicates (cues) and groups (Saurí, 2008).

but with reference also to Saurí’s (2008) dissertation for cues that are common in Twitter. The Porter Stemmer is applied to match inflections, e.g. *denies/denied*; for irregular cases not handled by the Porter Stemmer (e.g., *forget/forgot*), we include both forms. We use the CMU Twitter Part-of-Speech Tagger (Owoputi et al., 2013) to select only instances in the verb sense. Figure 2 shows the distribution of the cues and Figure 3 shows the distribution of the cue groups. For cues that appear in multiple groups, we chose the most common group.

- We run the Stanford Dependency parser to obtain labeled dependencies (De Marneffe et al., 2006), requiring that the cue has outgoing edges of the type NSUBJ (noun subject) and CCOMP (clausal complement). The subtree headed by the modifier of the CCOMP relation is considered the **claim**; the subtree headed by the modifier of the NSUBJ relation is considered the **source**. See Figure 4 for an example.
- We use a combination of regular expressions and dependency rules to capture expressions of the type “CLAIM, *according to* SOURCE.” Specifically, the PCOMP path from *according* is searched for the pattern *according to **. The text that matches the *** is the source and the remaining text other than the source is taken as the claim.
- Finally, we restrict consideration to tweets in which the source contains a named entity or twitter username. This eliminates expressions of personal belief such as *I doubt Obama will win*, as well as anonymous sources such as *Team sources report that LeBron has demanded a trade to New York*. Investigating the factuality judgments formed in response to such tweets is clearly an important problem for future research, but is outside the scope of this paper.

This heuristic pipeline may miss many relevant tweets, but since the overall volume is high, we

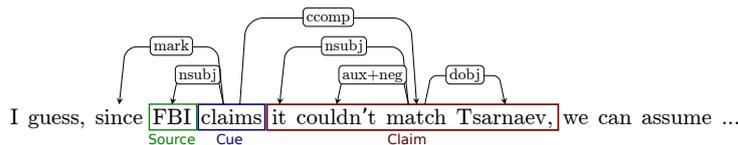


Figure 4: Dependency parse of an example message, with claim, source, and cue.

Total journalists	443
Total U.S. political journalists	103
Total tweets	959754
Tweets with cues	172706
Tweets with source and claims	40615
Total tweets annotated	1265
Unique sources in annotated dataset	766
Unigrams in annotated dataset	1345

Table 2: Count Statistics of the entire data collected and the annotated dataset



Figure 5: Turk annotation interface

prioritize precision. The resulting dataset is summarized in Table 2.

3 Annotation

We used Amazon Mechanical Turk (AMT) to collect ratings of claims. AMT has been widely used by the NLP community to collect data (Snow et al., 2008), with “best practices” defined to help requesters best design Turk jobs (Callison-Burch and Dredze, 2010). We followed these guidelines to perform pilot experiments to test the instruction set and the quality of responses. Based on the pilot study we designed Human Intelligence Tasks (HITs) to annotate 1265 claims.

Each HIT contained a batch of ten tweets and rewarded \$0.10 per hit. To ensure quality control we required the Turkers to have at least 85% hit approval rating and to reside in the United States, because the Twitter messages in our dataset were related to American politics. For each tweet,

we obtained five independent ratings from Turkers satisfying the above qualifications. The ratings were based on a 5-point Likert scale ranging from “[−2] Certainly False” to “[2] Certainly True” and allowing for “[0] Uncertain”. We also allowed for “Not Applicable” option to capture ratings where the Turkers did not have sufficient knowledge about the statement or if the statement was not really a claim. Figure 6 shows the set of instructions provided to the Turkers, and Figure 5 illustrates the annotation interface.²

We excluded tweets for which three or more Turkers gave a rating of “Not Applicable,” leaving us with a dataset of 1170 tweets. Within this set, the average variance per tweet (excluding “Not Applicable” ratings) was 0.585.

4 Modeling factuality judgments

Having obtained a corpus of factuality ratings, we now model the factors that drive these ratings.

4.1 Predictive accuracy

First, we attempt to determine the impact of various predictive features on rater judgments of factuality. We consider the following features:

- **Cue word:** after stemming
- **Cue word group:** as given in Table 1
- **Source:** represented by the named entity or username in the source field (see Figure 4)
- **Journalist:** represented by their Twitter ID
- **Claim:** represented by a bag-of-words vector from the claim field (Figure 4)

These features are used as predictors in a series of linear ridge regressions, where the dependent variable is the mean certainty rating. We throw out tweets that were rated as “not applicable” by a majority of raters, but otherwise ignore “not applicable” ratings of the remaining tweets. The goal of these regressions is to determine which features are predictive of raters’ factuality judgments. The ridge regression regularization parameter was tuned via cross-validation in the training set. We used the bootstrap to obtain multiple training/test

²The data is available at <https://www.github.com/jacobeisenstein/twitter-certainty>.

Rating Scale Instructions:

Before you get started, this introduction will explain what the different scores on the scale are supposed to reflect.

- **[-2] Certainly False** - You are certain that the claim is false.
- **[-1] Probably False** - You think that the claim might be false
- **[0] Uncertain (or Doubtful)** - The truth value of the claim is unknowable from the information presented.
- **[1] Probably True** - You think that the claim might be true
- **[2] Certainly True** - You are certain that the claim is true
- **NOT APPLICABLE** A statement falls under this category if any of the following condition is true:
 - The claim doesn't really have a truth value (e.g. "Huntsman says Ted Cruz should have stood up to the questioner...")
 - The statement doesn't look like a claim
 - You do not have sufficient knowledge to rate the statement.

Figure 6: User instructions for the annotation task

Features	Error
Baseline	.442
Cue word	.404*
Cue word group	.42
Source	.447
Journalist	.444
Claim	.476
Cue word + cue word group	.404*
All features	.420

Table 3: Linear regression error rates for each feature group. * indicates improvement over the baseline at $p < .05$.

splits (70% training), which were used for significance testing.

Table 3 reports mean average error for each feature group, as well as a baseline that simply reports the mean rating across the training set. Each accuracy was compared with the baseline using a paired z-test. Only the cue word features pass this test at $p < .05$. The other features do not help, even in combination with the cue word.

While these findings must be interpreted with caution, they suggest that readers — at least, Mechanical Turk workers — use relatively little independent judgment to assess the validity of quoted text that they encounter on Twitter. Of course, richer linguistic models, more advanced machine learning, or experiments with more carefully-selected readers might offer a different view. But the results at hand are most compatible with the conclusion that readers base their assessments of factuality only on the framing provided by the journalist who reports the quote.

4.2 Cue words and cue groups

Given the importance of cue words as a signal for factuality, we want to assess the factuality judgments induced by each cue. A second question is whether proposed groupings of cue words into groups cohere with such perceptions. Saurí (2008) describes several classes of source-

introducing predicates, which indicate how the source relates to the quoted claim. These classes are summarized in Table 1, along with frequently-occurring cues from our corpus. We rely on FactBank to assign the cue words to classes; the only word not covered by FactBank was *sense*, which we placed in predicates of perception.

We performed another set of linear regressions, again using the mean certainty rating as the dependent variable. In this case, there was no training/test split, so confidence intervals on the resulting parameters are computed using the analytic closed form. We performed two such regressions: first using only the individual cues as predictors, and then using only the cue groups. Results are shown in Figures 7 and 8; Figure 7 includes only cues which appear at least ten times, although all cues were included in the regression.

The cues that give the highest factuality coefficients are *learn* and *admit*, which are labeled as predicates of knowledge. These cues carry a substantial amount of framing, as they purport to describe the private mental state of the source. The word *admit* often applies to statements that are perceived as damaging to the source, such as *Bill Gates admits Control-Alt-Delete was a mistake*; since there can be no self-interest behind such statements, they may be perceived as more likely to be true.

Several of the cues with the lowest factuality coefficients are predicates of belief: *suggest*, *predict* and *think*. The words *suggest*, *think*, and *believe* also purport to describe the private mental state of the source, but their framing function is the opposite of the predicates of knowledge: they imply that it is important to mark the claim as the source's belief, and not a widely-accepted fact. For example, *Mubarak clearly believes he has the military leadership's support*.

A third group of interest are the predicates of report, which have widely-varying certainty coefficients. The cues *according*, *report*, *say*, and *tell*

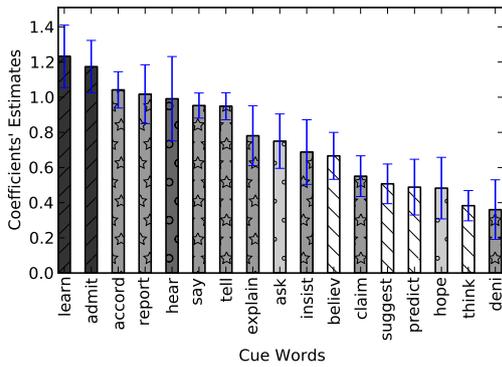


Figure 7: Linear regression coefficients for frequently-occurring cue words. Each word is patterned according to its group, shown in Figure 8.

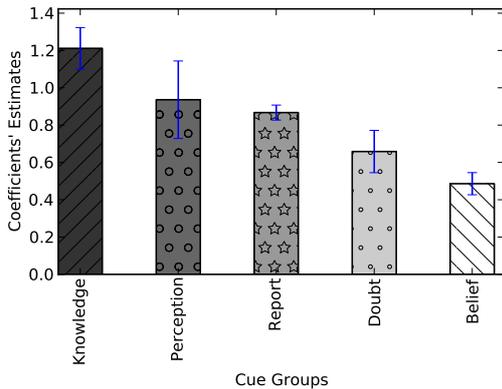


Figure 8: Linear regression coefficients for cue word group.

are strongly predictive of certainty, but the cues *claim* and *deny* convey uncertainty. Both *according* and *report* are often used in conjunction with impersonal and institutional sources, e.g., *Cucinelli trails McAuliffe by 24 points, according to a new poll*. In contrast, *insist*, *claim*, and *deny* imply that there is uncertainty about the quoted statement, e.g., *Christie insists that Fort Lee Mayor was never on my radar*. In this case, the fact that the predicate indicates a report is not enough to determine the framing: different sorts of reports carry radically different perceptions of factuality.

5 Related work

Factuality and Veridicality The creation of FactBank (Saurí and Pustejovsky, 2009) has enabled recent work on the factuality (or “veridicality”) of event mentions in text. Saurí and Pustejovsky (2012) propose a two-dimensional factuality annotation scheme, including polarity and certainty; they then build a classifier to predict annotations of factuality from statements in FactBank. Their work on source-introducing predicates provides part of the foundation for this re-

search, which focuses on quoted statements in social media text. de Marneffe et al. (2012) conduct an empirical evaluation of FactBank ratings from Mechanical Turk workers, finding a high degree of disagreement between raters. They also construct a statistical model to predict these ratings. We are unaware of prior work comparing the contribution of linguistic and extra-linguistic predictors (e.g., source and journalist features) for factuality ratings. This prior work also does not measure the impact of individual cues and cue classes on assessment of factuality.

Credibility in social media Recent work in the area of computational social science focuses on understanding credibility cues on Twitter. Such studies have found that users express concern over the credibility of tweets belonging to certain topics (politics, news, emergency). By manipulating several features of a tweet, Morris et al. (2012) found that in addition to content, users often use additional markers while assessing the tweet credibility, such as the user name of the source. The search for reliable signals of information credibility in social media has led to the construction of automatic classifiers to identify credible tweets (Castillo et al., 2011). However, this prior work has not explored the *linguistic* basis of factuality judgments, which we show to depend on framing devices such as cue words.

6 Conclusion

Perceptions of the factuality of quoted content are influenced by the cue words used to introduce them, while extra-linguistic factors, such as the source and the author, did not appear to be relevant in our experiments. This result is obtained from real tweets written by journalists; a natural counterpart study would be to experimentally manipulate this framing to see if the same perceptions apply. Another future direction would be to test whether the deployment of cue words as framing devices reflects the ideology of the journalist. We are also interested to group multiple instances of the same quote (Leskovec et al., 2009), and examine how its framing varies across different news outlets and over time.

Acknowledgments: This research was supported by DARPA-W911NF-12-1-0043 and by a Computational Journalism research award from Google. We thank the reviewers for their helpful feedback.

References

- Chris Callison-Burch and Mark Dredze. 2010. Creating speech and language data with amazon’s mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk*, pages 1–12. Association for Computational Linguistics.
- Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684. ACM.
- Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.
- Marie C. de Marneffe, Christopher D. Manning, and Christopher Potts. 2012. Did it happen? the pragmatic complexity of veridicality assessment. *Comput. Linguist.*, 38(2):301–333, June.
- Stephan Greene and Philip Resnik. 2009. More than words: Syntactic packaging and implicit sentiment. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 503–511, Boulder, Colorado, June. Association for Computational Linguistics.
- Alfred Hermida, Seth C Lewis, and Rodrigo Zamith. 2012. Sourcing the arab spring: A case study of andy carvins sources during the tunisian and egyptian revolutions. In *international symposium on online journalism, Austin, TX, April*, pages 20–21.
- Jure Leskovec, Lars Backstrom, and Jon Kleinberg. 2009. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506. ACM.
- Wei-Hao Lin, Theresa Wilson, Janyce Wiebe, and Alexander Hauptmann. 2006. Which side are you on?: Identifying perspectives at the document and sentence levels. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X ’06, pages 109–116, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Gilad Lotan, Erhardt Graeff, Mike Ananny, Devin Gaffney, Ian Pearce, et al. 2011. The arab spring—the revolutions were tweeted: Information flows during the 2011 tunisian and egyptian revolutions. *International Journal of Communication*, 5:31.
- Meredith Ringel Morris, Scott Counts, Asta Roseway, Aaron Hoff, and Julia Schwarz. 2012. Tweeting is believing?: understanding microblog credibility perceptions. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, pages 441–450. ACM.
- Tim O’Keefe, Silvia Pareti, James R Curran, Irena Koprinska, and Matthew Honnibal. 2012. A sequence labelling approach to quote attribution. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 790–799. Association for Computational Linguistics.
- Olutobi Owoputi, Brendan OConnor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of NAACL-HLT*, pages 380–390.
- Silvia Pareti. 2012. A database of attribution relations. In *LREC*, pages 3213–3217.
- Vahed Qazvinian, Emily Rosengren, Dragomir R Radev, and Qiaozhu Mei. 2011. Rumor has it: Identifying misinformation in microblogs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1589–1599. Association for Computational Linguistics.
- Roser Saurí and James Pustejovsky. 2009. Factbank: A corpus annotated with event factuality. *Language Resources and Evaluation*, 43(3):227–268.
- Roser Saurí and James Pustejovsky. 2012. Are you sure that this happened? assessing the factuality degree of events in text. *Comput. Linguist.*, 38(2):261–299, June.
- Roser Saurí. 2008. *A Factuality Profiler for Eventualities in Text*. Ph.D. thesis, Brandeis University.
- Rion Snow, Brendan O’Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP ’08, pages 254–263, Stroudsburg, PA, USA. Association for Computational Linguistics.

A Topic Model for Building Fine-grained Domain-specific Emotion Lexicon

Min Yang[‡] Baolin Peng[§] Zheng Chen[§] Dingju Zhu^{*†,¶} Kam-Pui Chow[‡]

[†]School of Computer Science, South China Normal University, Guangzhou, China
dingjuzhu@gmail.com

[‡]Department of Computer Science, The University of Hong Kong, Hong Kong
{myang, chow}@cs.hku.hk

[§]Department of Computer Science, Beihang University, Beijing, China
b.peng@cse.buaa.edu.cn, tzchen86@gmail.com

[¶]Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China

Abstract

Emotion lexicons play a crucial role in sentiment analysis and opinion mining. In this paper, we propose a novel Emotion-aware LDA (EaLDA) model to build a domain-specific lexicon for predefined emotions that include anger, disgust, fear, joy, sadness, surprise. The model uses a minimal set of domain-independent seed words as prior knowledge to discover a domain-specific lexicon, learning a fine-grained emotion lexicon much richer and adaptive to a specific domain. By comprehensive experiments, we show that our model can generate a high-quality fine-grained domain-specific emotion lexicon.

1 Introduction

Due to the popularity of opinion-rich resources (e.g., online review sites, forums, blogs and the microblogging websites), automatic extraction of opinions, emotions and sentiments in text is of great significance to obtain useful information for social and security studies. Various opinion mining applications have been proposed by different researchers, such as question answering, opinion mining, sentiment summarization, etc. As the fine-grained annotated data are expensive to get, the unsupervised approaches are preferred and more used in reality. Usually, a high quality emotion lexicon play a significant role when apply the unsupervised approaches for fine-grained emotion classification.

Thus far, most lexicon construction approaches focus on constructing general-purpose emotion lexicons (Stone et al., 1966; Hu and Liu, 2004; Wilson et al., 2005; Dong and Dong, 2006). However, since a specific word can carry various emotions in different domains, a general-purpose emotion lexicon is less accurate and less informative than a domain-specific lexicon (Baccianella et al., 2010). In addition, in previous work, most of the lexicons label the words on coarse-grained dimensions (positive, negative and neutrality). Such lexicons cannot accurately reflect the complexity of human emotions and sentiments. Lastly, previous emotion lexicons are mostly annotated based on many manually constructed resources (e.g., emotion lexicon, parsers, etc.). This limits the applicability of these methods to a broader range of tasks and languages.

To meet the challenges mentioned above, we propose a novel EaLDA model to construct a domain-specific emotion lexicon consisting of six primary emotions (i.e., anger, disgust, fear, joy, sadness and surprise). The proposed EaLDA model extends the standard Latent Dirichlet Allocation (LDA) (Blei et al., 2003) model by employing a small set of seeds to guide the model generating topics. Hence, the topics consequently group semantically related words into a same emotion category. The lexicon is thus able to best meet the user's specific needs. Our approach is a weakly supervised approach since only some seeds emotion sentiment words are needed to launch the process of lexicon construction. In practical applications, asking users to provide some seeds is easy as they usually have a good knowledge what are important in their domains.

*Dingju Zhu is the corresponding author

Extensive experiments are carried out to evaluate our model both qualitatively and quantitatively using benchmark dataset. The results demonstrate that our EaLDA model improves the quality and the coverage of state-of-the-art fine-grained lexicon.

2 Related Work

Emotion lexicon plays an important role in opinion mining and sentiment analysis. In order to build such a lexicon, many researchers have investigated various kinds of approaches. However, these methods could roughly be classified into two categories in terms of the used information. The first kind of approaches is based on thesaurus that utilizes synonyms or glosses to determine the sentiment orientation of a word. The availability of the WordNet (Miller, 1995) database is an important starting point for many thesaurus-based approaches (Kamps et al., 2004; Hu and Liu, 2004; Esuli and Sebastiani, 2006). The second kind of approaches is based on an idea that emotion words co-occurring with each others are likely to convey the same polarity. There are numerous studies in this field (Turney and Littman, 2003; Wiebe and Riloff, 2005; Esuli and Sebastiani, 2006; Barbosa and Feng, 2010).

Most of the previous studies for emotion lexicon construction are limited to positive and negative emotions. Recently, to enhance the increasingly emotional data, a few researches have been done to identify the fine-grained emotion of words (Strapparava and Mihalcea, 2007; Gill et al., 2008; Rao et al., 2012). For example, Gill et al. (2008) utilize computational linguistic tools to identify the emotions of the words (such as, joy, sadness, acceptance, disgust, fear, anger, surprise and anticipation). While, this approach is mainly for public use in general domains. Rao et al. (2012) propose an method of automatically building the word-emotion mapping dictionary for social emotion detection. However, the emotion lexicon is not outputted explicitly in this paper, and the approach is fully unsupervised which may be difficult to be adjusted to fit the personalized data set.

Our approach relates most closely to the method proposed by Xie and Li (2012) for the construction of lexicon annotated for polarity based on LDA model. Our approach differs from (Xie and Li, 2012) in two important ways: first, we do not address the task of polarity lexicon construction, but

instead we focus on building fine-grained emotion lexicon. Second, we don't assume that every word in documents is subjective, which is impractical in real world corpus.

3 Algorithm

In this section, we rigorously define the emotion-aware LDA model and its learning algorithm. We describe with the model description, a Gibbs sampling algorithm to infer the model parameters, and finally how to generate a emotion lexicon based on the model output.

3.1 Model Description

Like the standard LDA model, EaLDA is a generative model. To prevent conceptual confusion, we use a superscript “(e)” to indicate variables related to emotion topics, and use a superscript “(n)” to indicate variables of non-emotion topics. We assume that each document has two classes of topics: M emotion topics (corresponding to M different emotions) and K non-emotion topics (corresponding to topics that are not associated with any emotion). Each topic is represented by a multinomial distribution over words. In addition, we assume that the corpus vocabulary consists of V distinct words indexed by $\{1, \dots, V\}$.

For emotion topics, the EaLDA model draws the word distribution from a biased Dirichlet prior $\text{Dir}(\beta_k^{(e)})$. The vector $\beta_k^{(e)} \in \mathbb{R}^V$ is constructed with $\beta_k^{(e)} := \gamma_0^{(e)}(1^V - \Omega_k) + \gamma_1^{(e)}\Omega_k$, for $k \in \{1, \dots, M\}$. $\Omega_{k,w} = 1$ if and only if word w is a seed word for emotion k , otherwise $\Omega_{k,w} = 0$. The scalars $\gamma_0^{(e)}$ and $\gamma_1^{(e)}$ are hyperparameters of the model. Intuitively, when $\gamma_1^{(e)} > \gamma_0^{(e)}$, the biased prior ensures that the seed words are more probably drawn from the associated emotion topic.

The generative process of word distributions for non-emotion topics follows the standard LDA definition with a scalar hyperparameter $\beta^{(n)}$.

For each word in the document, we decide whether its topic is an emotion topic or a non-emotion topic by flipping a coin with head-tail probability $(p^{(e)}, p^{(n)})$, where $(p^{(e)}, p^{(n)}) \sim \text{Dir}(\alpha)$. The emotion (or non-emotion) topic is sampled according to a multinomial distribution $\text{Mult}(\theta^{(e)})$ (or $\text{Mult}(\theta^{(n)})$). Here, both $\theta^{(e)}$ and $\theta^{(n)}$ are document-level latent variables. They are generated from Dirichlet priors $\text{Dir}(\alpha^{(e)})$ and $\text{Dir}(\alpha^{(n)})$ with $\alpha^{(s)}$ and $\alpha^{(n)}$ being hyperparameters.

$$\mathbb{E}[\theta_i^{(e)}] = \frac{\alpha^{(e)} + m_i^{(e)}}{M\alpha^{(e)} + \sum_{i'=1}^M m_{i'}^{(e)}} \quad (6)$$

$$\mathbb{E}[\theta_j^{(n)}] = \frac{\alpha^{(n)} + m_j^{(n)}}{K\alpha^{(n)} + \sum_{j'=1}^K m_{j'}^{(n)}} \quad (7)$$

$$\mathbb{E}[\phi_{i,w}^{(e)}] = \frac{\beta_{i,w}^{(e)} + n_{i,w}^{(e)}}{\sum_{w'=1}^V (\beta_{i,w'}^{(e)} + n_{i,w'}^{(e)})} \quad (8)$$

$$\mathbb{E}[\phi_{j,w}^{(n)}] = \frac{\beta_{j,w}^{(n)} + n_{j,w}^{(n)}}{V\beta^{(n)} + \sum_{w'=1}^V n_{j,w'}^{(n)}} \quad (9)$$

Using the above equations, we can sample the topic z for each word iteratively and estimate all latent random variables.

3.3 Constructing Emotion Lexicon

Our final step is to construct the domain-specific emotion lexicon from the estimates $\phi^{(e)}$ and $\phi^{(n)}$ that we obtained from the EaLDA model.

For each word w in the vocabulary, we compare the $M + 1$ values $\{\phi_{1,w}^{(e)}, \dots, \phi_{M,w}^{(e)}\}$ and $\frac{1}{K} \sum_{i=1}^K \phi_{i,w}^{(n)}$. If $\phi_{i,w}^{(e)}$ is the largest, then the word w is added to the emotion dictionary for the i th emotion. Otherwise, $\frac{1}{K} \sum_{i=1}^K \phi_{i,w}^{(n)}$ is the largest among the $M + 1$ values, which suggests that the word w is more probably drawn from a non-emotion topic. Thus, the word is considered neutral and not included in the emotion dictionary.

4 Experiments

In this section, we report empirical evaluations of our proposed model. Since there is no metric explicitly measuring the quality of an emotion lexicon, we demonstrate the performance of our algorithm in two ways: (1) we perform a case study for the lexicon generated by our algorithm, and (2) we compare the results of solving emotion classification task using our lexicon against different methods, and demonstrate the advantage of our lexicon over other lexicons and other emotion classification systems.

4.1 Datasets

We conduct experiments to evaluate the effectiveness of our model on SemEval-2007 dataset. This is an gold-standard English dataset used in the 14th task of the SemEval-2007 workshop which focuses on classification of emotions in the text. The attributes include the news headlines, the score of

emotions of anger, disgust, fear, joy, sad and surprise normalizing from 0 to 100. Two data sets are available: a training data set consisting of 250 records, and a test data set with 1000 records. Following the strategy used in (Strapparava and Mihalcea, 2007), the task was carried out in an unsupervised setting for experiments.

In experiments, data preprocessing is performed on the data set. First, the texts are tokenized with a natural language toolkit NLTK¹. Then, we remove non-alphabet characters, numbers, pronoun, punctuation and stop words from the texts. Finally, Snowball stemmer² is applied so as to reduce the vocabulary size and settle the issue of data sparseness.

4.2 Emotion Lexicon Construction

We first settle down the implementation details for the EaLDA model, specifying the hyperparameters that we choose for the experiment. We set topic number $M = 6$, $K = 4$, and hyperparameters $\alpha = 0.75$, $\alpha^{(e)} = \alpha^{(n)} = 0.45$, $\beta^{(n)} = 0.5$. The vector $\beta^{(e)}$ is constructed from the seed dictionary using $\gamma = (0.25, 0.95)$.

As mentioned, we use a few domain-independent seed words as prior information for our model. To be specific, the seed words list contains 8 to 12 emotional words for each of the six emotion categories.³ However, it is important to note that the proposed models are flexible and do not need to have seeds for every topic.

Example words for each emotion generated from the SemEval-2007 dataset are reported in Table 1. The judgment is to some extent subjective. What we reported here are based on our judgments what are appropriate and what are not for each emotion topic. From Table 1, we observe that the generated words are informative and coherent. For example, the words “flu” and “cancer” are seemingly neutral by its surface meaning, actually expressing fear emotion for SemEval dataset. These domain-specific words are mostly not included in any other existing general-purpose emotion lexicons. The experimental results show that our algorithm can successfully construct a fine-grained domain-specific emotion lexicon for this corpus that is able to understand the connotation of the words that may not be obvious without the context.

¹<http://www.nltk.org>

²<http://snowball.tartarus.org/>

³<http://minyang.me/acl2014/seed-words.html>

Anger	Disgust	Fear	Joy	Sadness	Surprise
attack	mar	terror	good	kill	surprise
warn	sex	troop	win	die	first
gunman	lebanon	flu	prize	kidnap	jump
baghdad	game	dead	victory	lose	marijuana
immigration	gaze	die	adopt	confuse	arrest
hit	cancer	cancer	madonna	crach	sweat
kidnap	amish	kidnap	celebrity	leave	find
kill	imigration	force	boost	cancer	attack
alzheim	sink	iraq	ship	flu	hiv
iraqi	force	fear	star	kidnap	discover

Table 1: Part of Emotion example words

Algorithm	Anger	Disgust	Fear	Joy	Sadness	Surprise
WordNet-Affect	6.06%	-	-	22.81%	17.31%	9.92%
SWAT	7.06%	-	18.27%	14.91%	17.44%	11.78%
UA	16.03%	-	20.06%	4.21%	1.76%	15.00%
UPAR7	3.02%	-	4.72%	11.87%	17.44%	15.00%
EaLDA	16.65%	10.52%	26.21%	25.57%	36.85%	20.17%

Table 2: Experiment results for emotion classification in term of F1 score

4.3 Document-level Emotion Classification

We compare the performance between a popular emotion lexicon WordNet-Affect (Strapparava and Valitutti, 2004) and our approach for emotion classification task. We also compare our results with those obtained by three systems participating in the SemEval-2007 emotion annotation task: SWAT, UPAR7 and UA. The emotion classification results is evaluated for each emotion category separately. For each emotion category, we evaluates it as a binary classification problem. In the evaluation of emotion lexicons, the binary classification is performed in a very simple way. For each emotion category and each text, we compare the number of words within this emotion category, and the average number of words within other emotion categories, to output a binary prediction of 1 or 0. This simple approach is chosen to evaluate the robustness of our emotion lexicon.

In the experiments, performance is evaluated in terms of F1-score. We summarize the results in Table 2. As an easy observation, the emotion lexicon generated by the EaLDA model consistently and significantly outperforms the WordNet-Affect emotion lexicon and other three emotion classification systems. In particular, we are able to obtain an overall F1-score of 10.52% for disgust classification task which is difficult to work out using pre-

viously proposed methods. The advantage of our model may come from its capability of exploring domain-specific emotions which include not only explicit emotion words, but also implicit ones.

5 Conclusions and Future Work

In this paper, we have presented a novel emotion-aware LDA model that is able to quickly build a fine-grained domain-specific emotion lexicon for languages without many manually constructed resources. The proposed EaLDA model extends the standard LDA model by accepting a set of domain-independent emotion words as prior knowledge, and guiding to group semantically related words into the same emotion category. Thus, it makes the emotion lexicon containing much richer and adaptive domain-specific emotion words. Experimental results showed that the emotional lexicons generated by our algorithm is of high quality, and can assist emotion classification task.

For future works, we hope to extend the proposed EaLDA model by exploiting discourse structure knowledge, which has been shown significant in identifying the polarity of content-aware words.

References

- Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*, volume 10, pages 2200–2204.
- Luciano Barbosa and Junlan Feng. 2010. Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 36–44. Association for Computational Linguistics.
- David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.
- Zhendong Dong and Qiang Dong. 2006. *HowNet and the Computation of Meaning*. World Scientific.
- Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422.
- Alastair J Gill, Robert M French, Darren Gergle, and Jon Oberlander. 2008. The language of emotion in short blog texts. In *CSCW*, volume 8, pages 299–302.
- Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.
- Jaap Kamps, MJ Marx, Robert J Mokken, and Maarten De Rijke. 2004. Using wordnet to measure semantic orientations of adjectives.
- George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.
- Yanghui Rao, Xiaojun Quan, Liu Wenyin, Qing Li, and Mingliang Chen. 2012. Building word-emotion mapping dictionary for online news. In *SDAD 2012 The 1st International Workshop on Sentiment Discovery from Affective Data*, page 28.
- Philip J Stone, Dexter C Dunphy, and Marshall S Smith. 1966. The general inquirer: A computer approach to content analysis.
- Carlo Strapparava and Rada Mihalcea. 2007. Semeval-2007 task 14: Affective text. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 70–74. Association for Computational Linguistics.
- Carlo Strapparava and Alessandro Valitutti. 2004. Wordnet affect: an affective extension of wordnet. In *LREC*, volume 4, pages 1083–1086.
- Peter D Turney and Michael L Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems (TOIS)*, 21(4):315–346.
- Janyce Wiebe and Ellen Riloff. 2005. Creating subjective and objective sentence classifiers from unannotated texts. In *Computational Linguistics and Intelligent Text Processing*, pages 486–497. Springer.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 347–354. Association for Computational Linguistics.
- Rui Xie and Chunping Li. 2012. Lexicon construction: A topic model approach. In *Systems and Informatics (ICSAI), 2012 International Conference on*, pages 2299–2303. IEEE.

DepecheMood: a Lexicon for Emotion Analysis from Crowd-Annotated News

Jacopo Staiano
University of Trento
Trento - Italy
staiano@disi.unitn.it

Marco Guerini
Trento RISE
Trento - Italy
marco.guerini@trentorise.eu

Abstract

While many lexica annotated with words polarity are available for sentiment analysis, very few tackle the harder task of emotion analysis and are usually quite limited in coverage. In this paper, we present a novel approach for extracting – in a totally automated way – a high-coverage and high-precision lexicon of roughly 37 thousand terms annotated with emotion scores, called *DepecheMood*. Our approach exploits in an original way ‘crowd-sourced’ affective annotation implicitly provided by readers of news articles from *rappler.com*. By providing new state-of-the-art performances in unsupervised settings for regression and classification tasks, even using a naïve approach, our experiments show the beneficial impact of harvesting social media data for affective lexicon building.

1 Introduction

Sentiment analysis has proved useful in several application scenarios, for instance in buzz monitoring – the marketing technique for keeping track of consumer responses to services and products – where identifying positive and negative customer experiences helps to assess product and service demand, tackle crisis management, etc.

On the other hand, the use of finer-grained models, accounting for the role of individual emotions, is still in its infancy. The simple division in ‘positive’ vs. ‘negative’ comments may not suffice, as in these examples: ‘*I’m so miserable, I dropped my iPhone in the water and now it’s not working anymore*’ (SADNESS) vs. ‘*I am very upset, my new iPhone keeps not working!*’ (ANGER). While both texts express a negative sentiment, the latter, connected to anger, is more relevant for buzz monitor-

ing. Thus, emotion analysis represents a natural evolution of sentiment analysis.

Many approaches to sentiment analysis make use of lexical resources – i.e. lists of positive and negative words – often deployed as baselines or as features for other methods, usually machine learning based (Liu and Zhang, 2012). In these lexica, words are associated with their prior polarity, i.e. whether such word out of context evokes something positive or something negative. For example, *wonderful* has a positive connotation – prior polarity – while *horrible* has a negative one.

The quest for a high precision and high coverage lexicon, where words are associated with either sentiment or emotion scores, has several reasons. First, it is fundamental for tasks such as affective modification of existing texts, where words’ polarity together with their score are necessary for creating multiple *graded* variations of the original text (Inkpen et al., 2006; Guerini et al., 2008; Whitehead and Cavedon, 2010).

Second, considering word order makes a difference in sentiment analysis. This calls for a role of compositionality, where the score of a sentence is computed by composing the scores of the words up in the syntactic tree. Works worth mentioning in this connection are: Socher et al. (2013), which uses recursive neural networks to learn compositional rules for sentiment analysis, and (Neviarouskaya et al., 2009; Neviarouskaya et al., 2011) which exploit hand-coded rules to compose the emotions expressed by words in a sentence. In this respect, compositional approaches represent a new promising trend, since all other approaches, either using semantic similarity or Bag-of-Words (BOW) based machine-learning, cannot handle, for example, cases of texts with same wording but different words order: “*The dangerous killer escaped one month ago, but recently he was arrested*” (RELIEF, HAPPYNESS) vs. “*The dangerous killer was arrested one month ago, but re-*

cently he escaped” (FEAR). The work in (Wang and Manning, 2012) partially accounts for this problem and argues that using word bigram features allows improving over BOW based methods, where words are taken as features in isolation. This way it is possible to capture simple compositional phenomena like polarity reversing in “killing cancer”.

Finally, tasks such as copywriting, where evocative names are a key element to a successful product (Ozbal and Strapparava, 2012; Ozbal et al., 2012) require exhaustive lists of emotion related words. In such cases no context is given and the brand name alone, with its perceived prior polarity, is responsible for stating the area of competition and evoking semantic associations. For example *Mitsubishi* changed the name of one of its SUVs for the Spanish market, since the original name *Pajero* had a very negative prior polarity, as it means ‘wanker’ in Spanish (Piller, 2003). Evoking emotions is also fundamental for a successful name: consider names of a perfume like *Obsession*, or technological products like *MacBook air*.

In this work, we aim at automatically producing a high coverage and high precision emotion lexicon using distributional semantics, with numerical scores associated with each emotion, like it has already been done for sentiment analysis. To this end, we take advantage in an original way of massive crowd-sourced affective annotations associated with news articles, obtained by crawling the `rappler.com` social news network. We also evaluate our lexicon by integrating it in unsupervised classification and regression settings for emotion recognition. Results indicate that the use of our resource, even if automatically acquired, is highly beneficial in affective text recognition.

2 Related Work

Within the broad field of sentiment analysis, we hereby provide a short review of research efforts put towards building sentiment and emotion lexica, regardless of the approach in which such lists are then used (machine learning, rule based or deep learning). A general overview can be found in (Pang and Lee, 2008; Liu and Zhang, 2012; Wilson et al., 2004; Paltoglou et al., 2010).

Sentiment Lexica. In recent years there has been an increasing focus on producing lists of words (lexica) with prior polarities, to be used in sentiment analysis. When building such lists, a

trade-off between coverage of the resource and its precision is to be found.

One of the most well-known resources is *Senti-WordNet* (SWN) (Esuli and Sebastiani, 2006; Baccianella et al., 2010), in which each entry is associated with the numerical scores $Pos(s)$ and $Neg(s)$, ranging from 0 to 1. These scores – automatically assigned starting from a bunch of seed terms – represent the positive and negative valence (or posterior polarity) of each entry, that takes the form `lemma#pos#sense-number`. Starting from SWN, several prior polarities for words (*SWN-prior*), in the form `lemma#Pos`, can be computed (e.g. considering only the first-sense, averaging on all the senses, etc.). These approaches, detailed in (Guerini et al., 2013), produce a list of 155k words, where the lower precision given by the automatic scoring of SWN is compensated by the high coverage.

Another widely used resource is *ANEW* (Bradley and Lang, 1999), providing valence scores for 1k words, which were manually assigned by several annotators. This resource has a low coverage, but the precision is maximized. Similarly, the *SO-CAL* entries (Taboada et al., 2011) were manually tagged by a small number of annotators with a multi-class label (from `very_negative` to `very_positive`). These ratings were further validated through crowd-sourcing, ending up with a list of roughly 4k words. More recently, a resource that replicated ANEW annotation approach using crowd-sourcing, was released (Warriner et al., 2013), providing sentiment scores for 14k words. Interestingly, this resource annotates the most frequent words in English, so, even if lexicon coverage is still far lower than *SWN-prior*, it grants a high coverage, with human precision, of language use.

Finally, the *General Inquirer* lexicon (Stone et al., 1966) provides a binary classification (`positive/negative`) of 4k sentiment-bearing words, while the resource in (Wilson et al., 2005) expands the *General Inquirer* to 6k words.

Emotion Lexica. Compared to sentiment lexica, far less emotion lexica have been produced, and all have lower coverage. One of the most used resources is *WordNetAffect* (Strapparava and Valitutti, 2004) which contains manually assigned affective labels to WordNet synsets (`ANGER`, `JOY`, `FEAR`, etc.). It currently provides 900 annotated synsets and 1.6k words in the form

	AFRAID	AMUSED	ANGRY	ANNOYED	DONT_CARE	HAPPY	INSPIRED	SAD
doc_10002	0.75	0.00	0.00	0.00	0.00	0.00	0.25	0.00
doc_10003	0.00	0.50	0.00	0.16	0.17	0.17	0.00	0.00
doc_10004	0.52	0.02	0.03	0.02	0.02	0.06	0.02	0.31
doc_10011	0.40	0.00	0.00	0.20	0.00	0.20	0.20	0.00
doc_10028	0.00	0.30	0.08	0.00	0.00	0.23	0.31	0.08

Table 1: An excerpt of the Document-by-Emotion Matrix - M_{DE}

lemma#PoS#sense, corresponding to roughly 1 thousand lemma#PoS.

AffectNet, part of the SenticNet project (Cambria and Hussain, 2012), contains 10k words (out of 23k entries) taken from ConceptNet and aligned with WordNetAffect. This resource extends WordNetAffect labels to concepts like ‘have breakfast’. *Fuzzy Affect Lexicon* (Subasic and Huettner, 2001) contains roughly 4k lemma#PoS manually annotated by one linguist using 80 emotion labels. *EmoLex* (Mohammad and Turney, 2013) contains almost 10k lemmas annotated with an intensity label for each emotion using Mechanical Turk. Finally *Affect database* is an extension of SentiFul (Neviarouskaya et al., 2007) and contains 2.5K words in the form lemma#PoS. The latter is the only lexicon providing words annotated also with emotion scores rather than only with labels.

3 Dataset Collection

To build our emotion lexicon we harvested all the news articles from `rappler.com`, as of June 3rd 2013: the final dataset consists of 13.5 M words over 25.3 K documents, with an average of 530 words per document. For each document, along with the text we also harvested the information displayed by Rappler’s *Mood Meter*, a small interface offering the readers the opportunity to click on the emotion that a given Rappler story made them feel. The idea behind the Mood Meter is actually “getting people to *crowdsource* the mood for the day”¹, and returning the percentage of votes for each emotion label for a given story. This way, hundreds of thousands votes have been collected since the launch of the service. In our novel approach to ‘crowdsourcing’, as compared to other NLP tasks that rely on tools like Amazon’s Mechanical Turk (Snow et al., 2008), the subjects are aware of the ‘implicit annotation task’ but they are not paid. From this data, we built a document-by-emotion matrix M_{DE} , providing the voting percentages for each document in the eight

¹<http://nie.mn/QuD17Z>

affective dimensions available in Rappler. An excerpt is provided in Table 1.

The idea of using documents annotated with emotions is not new (Strapparava and Mihalcea, 2008; Mishne, 2005; Bellegarda, 2010), but these works had the limitation of providing a single emotion label per document, rather than a score for each emotion, and, moreover, the annotation was performed by the author of the document alone.

Table 2 reports the average percentage of votes for each emotion on the whole corpus: HAPPINESS has a far higher percentage of votes (at least three times). There are several possible explanations, out of the scope of the present paper, for this bias: (i) it is due to cultural characteristics of the audience (ii) the bias is in the dataset itself, being formed mainly by ‘positive’ news; (iii) it is a psychological phenomenon due to the fact that people tend to express more positive moods on social networks (Quercia et al., 2011; Vittengl and Holt, 1998; De Choudhury et al., 2012). In any case, the predominance of happy mood has been found in other datasets, for instance `LiveJournal.com` posts (Strapparava and Mihalcea, 2008). In the following section we will discuss how we handled this problem.

EMOTION	Votes _{μ}	EMOTION	Votes _{μ}
AFRAID	0.04	DONT_CARE	0.05
AMUSED	0.10	HAPPY	0.32
ANGRY	0.10	INSPIRED	0.10
ANNOYED	0.06	SAD	0.11

Table 2: Average percentages of votes.

4 Emotion Lexicon Creation

As a next step we built a word-by-emotion matrix starting from M_{DE} using an approach based on compositional semantics. To do so, we first lemmatized and PoS tagged all the documents (where PoS can be adj., nouns, verbs, adv.) and kept only those lemma#PoS present also in WordNet, similar to SWN-prior and WordNetAffect resources, to which we want to align. We then computed the term-by-document matrices using raw

Word	AFRAID	AMUSED	ANGRY	ANNOYED	DONT_CARE	HAPPY	INSPIRED	SAD
awe#n	0.08	0.12	0.04	0.11	0.07	0.15	0.38	0.05
comical#a	0.02	0.51	0.04	0.05	0.12	0.17	0.03	0.06
crime#n	0.11	0.10	0.23	0.15	0.07	0.09	0.09	0.15
criminal#a	0.12	0.10	0.25	0.14	0.10	0.11	0.07	0.11
dead#a	0.17	0.07	0.17	0.07	0.07	0.05	0.05	0.35
funny#a	0.04	0.29	0.04	0.11	0.16	0.13	0.15	0.08
future#n	0.09	0.12	0.09	0.12	0.13	0.13	0.21	0.10
game#n	0.06	0.15	0.06	0.08	0.15	0.23	0.15	0.12
kill#v	0.23	0.06	0.21	0.07	0.05	0.06	0.05	0.27
rapist#n	0.02	0.07	0.46	0.07	0.08	0.16	0.03	0.12
sad#a	0.06	0.12	0.09	0.14	0.13	0.07	0.15	0.24
warning#n	0.44	0.06	0.09	0.09	0.06	0.06	0.04	0.16

Table 3: An excerpt of the Word-by-Emotion Matrix (M_{WE}) using normalized frequencies (nf). Emotions weighting more than 20% in a word are highlighted for readability purposes.

frequencies, normalized frequencies, and tf-idf ($M_{WD,f}$, $M_{WD,nf}$ and $M_{WD,tfidf}$ respectively), so to test which of the three weights is better. After that, we applied matrix multiplication between the document-by-emotion and word-by-document matrices ($M_{DE} \cdot M_{WD}$) to obtain a (raw) word-by-emotion matrix M_{WE} . This method allows us to ‘merge’ words with emotions by summing the products of the weight of a word with the weight of the emotions in each document.

Finally, we transformed M_{WE} by first applying normalization column-wise (so to eliminate the over representation for happiness as discussed in Section 3) and then scaling the data row-wise so to sum up to one. An excerpt of the final Matrix M_{WE} is presented in Table 3, and it can be interpreted as a list of words with scores that represent how much weight a given word has in the affective dimensions we consider. So, for example, `awe#n` has a predominant weight in INSPIRED (0.38), `comical#a` has a predominant weight in AMUSED (0.51), while `kill#v` has a predominant weight in AFRAID, ANGRY and SAD (0.23, 0.21 and 0.27 respectively). This matrix, that we call `DepecheMood`², represents our emotion lexicon, it contains 37k entries and is freely available for research purposes at <http://git.io/MqyoIg>.

5 Experiments

To evaluate the performance we can obtain with our lexicon, we use the public dataset provided for the SemEval 2007 task on ‘Affective Text’ (Straparava and Mihalcea, 2007). The task was focused on emotion recognition in one thousand news headlines, both in regression and classification settings. Headlines typically consist of a few

²In French, ‘depeche’ means dispatch/news.

words and are often written with the intention to ‘provoke’ emotions so to attract the readers’ attention. An example of headline from the dataset is the following: “*Iraq car bombings kill 22 People, wound more than 60*”. For the regression task the values provided are: `<anger (0.32), disgust (0.27), fear (0.84), joy (0.0), sadness (0.95), surprise (0.20)>` while for the classification task the labels provided are `{FEAR, SADNESS}`.

This dataset is of interest to us since the ‘compositional’ problem is less prominent given the simplified syntax of news headlines, containing, for example, fewer adverbs (like negations or intensifiers) than normal sentences (Turchi et al., 2012). Furthermore, this is to our knowledge the only dataset available providing numerical scores for emotions. Finally, this dataset was meant for unsupervised approaches (just a small trial sample was provided), so to avoid simple text categorization approaches.

As the affective dimensions present in the test set – based on the six basic emotions model (Ekman and Friesen, 1971) – do not exactly match with the ones provided by Rappler’s Mood Meter, we first define a mapping between the two when possible, see Table 4. Then, we proceed to transform the test headlines to the `lemma#PoS` format.

SemEval	Rappler	SemEval	Rappler
FEAR	AFRAID	SURPRISE	INSPIRED
ANGER	ANGRY	-	ANNOYED
JOY	HAPPY	-	AMUSED
SADNESS	SAD	-	DON’T CARE

Table 4: Mapping of Rappler labels on SemEval2007. In bold, cases of suboptimal mapping.

Only one test headline contained exclusively words not present in `DepecheMood`, further indi-

cating the high-coverage nature of our resource. In Table 5 we report the coverage of some Sentiment and Emotion Lexica of different sizes on the same dataset. Similar to Warriner et al. (2013), we observe that even if the number of entries of our lexicon is far lower than SWN-prior approaches, the fact that we extracted and annotated words from documents grants a high coverage of language use.

Sentiment Lexica	ANEW	1k entries	0.10
	Warriner et. al	13k entries	0.51
	SWN-prior	155k entries	0.67
Emotion Lexica	WNAffect	1k entries	0.12
	DepecheMood	37k entries	0.64

Table 5: Statistics on words coverage per headline.

Since our primary goal is to assess the quality of DepecheMood we first focus on the regression task. We do so by using a very naïve approach, similar to “WordNetAffect presence” discussed in (Strapparava and Mihalcea, 2008): for each headline, we simply compute a value, for any affective dimension, by averaging the corresponding affective scores –obtained from DepecheMood– of all lemma#PoS present in the headline.

In Table 6 we report the results obtained using the three versions of our resource (Pearson correlation), along with the best performance on each emotion of other systems³ ($best_{se}$); the last column contains the upper bound of inter-annotator agreement. For all the 5 emotions we improve over the best performing systems (DISGUST has no alignment with our labels and was discarded).

Interestingly, even using a sub-optimal alignment for SURPRISE we still manage to outperform other systems. Considering the naïve approach we used, we can reasonably conclude that the quality and coverage of our resource are the reason of such results, and that adopting more complex approaches (i.e. compositionality) can possibly further improve performances in text-based emotion recognition.

As a final test, we evaluate our resource in the classification task. The naïve approach used in this case consists in mapping the average of the scores of all words in the headline to a binary decision with fixed threshold at 0.5 for each emotion (after min-max normalization on all test headlines

³Systems participating in the ‘Affective Text’ task plus the approaches in (Strapparava and Mihalcea, 2008). Other supervised approaches in the classification task (Mohammad, 2012; Bellegarda, 2010; Chaffar and Inkpen, 2011), reporting only overall performances, are not considered.

	DepecheMood			$best_{se}$	upper
	f	nf	$tfidf$		
FEAR	0.56	0.54	0.53	0.45	0.64
ANGER	0.36	0.38	0.36	0.32	0.50
SURPRISE*	0.25	0.21	0.24	0.16	0.36
JOY	0.39	0.40	0.39	0.26	0.60
SADNESS	0.48	0.47	0.46	0.41	0.68

Table 6: Regression results – Pearson’s correlation

scores). In Table 7 we report the results (F1 measure) of our approach along with the best performance of other systems on each emotion ($best_{se}$), as in the previous case. For 3 emotions out of 5 we improve over the best performing systems, for one emotion we obtain the same results, and for one emotion we do not outperform other systems. In this case the difference in performances among the various ways of representing the word-by-document matrix is more prominent: normalized frequencies (nf) provide the best results.

	DepecheMood			$best_{se}$
	f	nf	$tfidf$	
FEAR	0.25	0.32	0.31	0.23
ANGER	0.00	0.00	0.00	0.17
SURPRISE*	0.13	0.16	0.09	0.15
JOY	0.22	0.30	0.32	0.32
SADNESS	0.36	0.40	0.38	0.30

Table 7: Classification results – F1 measures

6 Conclusions

We presented DepecheMood, an emotion lexicon built in a novel and totally automated way by harvesting crowd-sourced affective annotation from a social news network. Our experimental results indicate high-coverage and high-precision of the lexicon, showing significant improvements over state-of-the-art unsupervised approaches even when using the resource with very naïve classification and regression strategies. We believe that the wealth of information provided by social media can be harnessed to build models and resources for emotion recognition from text, going a step beyond sentiment analysis. Our future work will include testing Singular Value Decomposition on the word-by-document matrices, allowing to propagate emotions values for a document to similar words non present in the document itself, and the study of perceived mood effects on virality indices and readers engagement by exploiting tweets, likes, reshares and comments.

This work has been partially supported by the Trento RISE PerTe project.

References

- S. Baccianella, A. Esuli, and F. Sebastiani. 2010. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the Conference on International Language Resources and Evaluation (LREC)*, pages 2200–2204, Valletta, Malta.
- J. R. Bellegarda. 2010. Emotion analysis using latent affective folding and embedding. In *Proceedings of the NAACL HLT 2010 workshop on computational approaches to analysis and generation of emotion in text*, pages 1–9. Association for Computational Linguistics.
- M. Bradley and P. Lang. 1999. Affective norms for english words (ANEW): Instruction manual and affective ratings. *Technical Report C-1, University of Florida*.
- E. Cambria and A. Hussain. 2012. *Sentic computing*. Springer.
- S. Chaffar and D. Inkpen. 2011. Using a heterogeneous dataset for emotion analysis in text. In *Advances in Artificial Intelligence*, pages 62–67. Springer.
- M. De Choudhury, S. Counts, and M. Gamon. 2012. Not all moods are created equal! exploring human emotional states in social media. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM)*.
- P. Ekman and W. V. Friesen. 1971. Constants across cultures in the face and emotion. *Journal of Personality and Social Psychology*, 17:124–129.
- A. Esuli and F. Sebastiani. 2006. SentiWordNet: A publicly available lexical resource for opinion mining. In *Proceedings of the Conference on International Language Resources and Evaluation (LREC)*, pages 417–422, Genova, IT.
- M. Guerini, O. Stock, and C. Strapparava. 2008. Valentino: A tool for valence shifting of natural language texts. In *Proceedings of the Conference on International Language Resources and Evaluation (LREC)*, Marrakech, Morocco.
- M. Guerini, L. Gatti, and M. Turchi. 2013. Sentiment analysis: How to derive prior polarities from sentiwordnet. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1259–1269.
- D. Z. Inkpen, O. Feiguina, and G. Hirst. 2006. Generating more-positive and more-negative text. In *Computing Attitude and Affect in Text: Theory and Applications*, pages 187–198. Springer.
- B. Liu and L. Zhang. 2012. A survey of opinion mining and sentiment analysis. *Mining Text Data*, pages 415–463.
- G. Mishne. 2005. Experiments with mood classification in blog posts. In *Proceedings of ACM SIGIR 2005 Workshop on Stylistic Analysis of Text for Information Access*, volume 19.
- S. M. Mohammad and P. D. Turney. 2013. Crowdsourcing a word–emotion association lexicon. *Computational Intelligence*, 29(3):436–465.
- S. M. Mohammad. 2012. # Emotional tweets. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics (*Sem)*, pages 246–255. Association for Computational Linguistics.
- A. Neviarouskaya, H. Prendinger, and M. Ishizuka. 2007. Textual affect sensing for sociable and expressive online communication. In A. Paiva, R. Prada, and R. Picard, editors, *Affective Computing and Intelligent Interaction*, volume 4738 of *Lecture Notes in Computer Science*, pages 218–229. Springer Berlin Heidelberg.
- A. Neviarouskaya, H. Prendinger, and M. Ishizuka. 2009. Compositionality principle in recognition of fine-grained emotions from text. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM)*.
- A. Neviarouskaya, H. Prendinger, and M. Ishizuka. 2011. Affect analysis model: novel rule-based approach to affect sensing from text. *Natural Language Engineering*, 17(1):95.
- G. Ozbal and C. Strapparava. 2012. A computational approach to the automation of creative naming. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*.
- G. Ozbal, C. Strapparava, and M. Guerini. 2012. Brand pitt: A corpus to explore the art of naming. In *Proceedings of the Conference on International Language Resources and Evaluation (LREC)*.
- G. Paltoglou, M. Thelwall, and K. Buckley. 2010. Online textual communications annotated with grades of emotion strength. In *Proceedings of the 3rd International Workshop of Emotion: Corpora for research on Emotion and Affect*, pages 25–31.
- B. Pang and L. Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.
- I. Piller. 2003. 10. advertising as a site of language contact. *Annual Review of Applied Linguistics*, 23:170–183.
- D. Quercia, J. Ellis, L. Capra, and J. Crowcroft. 2011. In the mood for being influential on twitter. *Proceedings of IEEE SocialCom'11*.
- R. Snow, B. O'Connor, D. Jurafsky, and A. Ng. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 254–263.

- R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642.
- P. Stone, D. Dunphy, and M. Smith. 1966. *The General Inquirer: A Computer Approach to Content Analysis*. MIT press.
- C. Strapparava and R. Mihalcea. 2007. Semeval-2007 task 14: Affective text. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 70–74. Association for Computational Linguistics.
- C. Strapparava and R. Mihalcea. 2008. Learning to identify emotions in text. In *Proceedings of the 2008 ACM symposium on Applied computing*, pages 1556–1560. ACM.
- C. Strapparava and A. Valitutti. 2004. WordNet-Affect: an affective extension of WordNet. In *Proceedings of the Conference on International Language Resources and Evaluation (LREC)*, pages 1083 – 1086, Lisbon, May.
- P. Subasic and A. Huettner. 2001. Affect analysis of text using fuzzy semantic typing. *Fuzzy Systems, IEEE Transactions on*, 9(4):483–496.
- M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307.
- M. Turchi, M. Atkinson, A. Wilcox, B. Crawley, S. Bucci, R. Steinberger, and E. Van der Goot. 2012. Onto: optima news translation system. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 25–30. Association for Computational Linguistics.
- J. R. Vittengl and C. S. Holt. 1998. A time-series diary study of mood and social interaction. *Motivation and Emotion*, 22(3):255–275.
- S. Wang and C. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*.
- A. B. Warriner, V. Kuperman, and M. Brysbaert. 2013. Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior research methods*, 45(4):1191–1207.
- S. Whitehead and L. Cavedon. 2010. Generating shifting sentiment for a conversational agent. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 89–97, Los Angeles, CA, June. Association for Computational Linguistics.
- T. Wilson, J. Wiebe, and R. Hwa. 2004. Just how mad are you? finding strong and weak opinion clauses. In *Proceedings of AAAI*, pages 761–769.
- T. Wilson, J. Wiebe, and P. Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 347–354.

Improving Twitter Sentiment Analysis with Topic-Based Mixture Modeling and Semi-Supervised Training

Bing Xiang *

IBM Watson
1101 Kitchawan Rd
Yorktown Heights, NY 10598, USA
bingxia@us.ibm.com

Liang Zhou

Thomson Reuters
3 Times Square
New York, NY 10036, USA
l.zhou@thomsonreuters.com

Abstract

In this paper, we present multiple approaches to improve sentiment analysis on Twitter data. We first establish a state-of-the-art baseline with a rich feature set. Then we build a topic-based sentiment mixture model with topic-specific data in a semi-supervised training framework. The topic information is generated through topic modeling based on an efficient implementation of Latent Dirichlet Allocation (LDA). The proposed sentiment model outperforms the top system in the task of *Sentiment Analysis in Twitter* in SemEval-2013 in terms of averaged F scores.

1 Introduction

Social media, such as Twitter and Facebook, has attracted significant attention in recent years. The vast amount of data available online provides a unique opportunity to the people working on natural language processing (NLP) and related fields. Sentiment analysis is one of the areas that has large potential in real-world applications. For example, monitoring the trend of sentiment for a specific company or product mentioned in social media can be useful in stock prediction and product marketing.

In this paper, we focus on sentiment analysis of Twitter data (tweets). It is one of the challenging tasks in NLP given the length limit on each tweet (up to 140 characters) and also the informal conversation. Many approaches have been proposed previously to improve sentiment analysis on Twitter data. For example, Nakov et al. (2013) provide an overview on the systems submitted to one of the SemEval-2013 tasks, *Sentiment Analysis in Twitter*. A variety of features have been utilized for

* This work was done when the author was with Thomson Reuters.

sentiment classification on tweets. They include lexical features (e.g. word lexicon), syntactic features (e.g. Part-of-Speech), Twitter-specific features (e.g. emoticons), etc. However, all of these features only capture local information in the data and do not take into account of the global higher-level information, such as topic information.

Two example tweets are given below, with the word “*offensive*” appearing in both of them.

- *Im gonna post something that might be **offensive** to people in Singapore.*
- *#FSU **offensive** coordinator Randy Sanders coached for Tennessee in 1st #BCS title game.*

Generally “*offensive*” is used as a negative word (as in the first tweet), but it bears no sentiment in the second tweet when people are talking about a football game. Even though some local contextual features could be helpful to distinguish the two cases above, they still may not be enough to get the sentiment on the whole message correct. Also, the local features often suffer from the sparsity problem. This motivates us to explore topic information explicitly in the task of sentiment analysis on Twitter data.

There exists some work on applying topic information in sentiment analysis, such as (Mei et al., 2007), (Branavan et al., 2008), (Jo and Oh, 2011) and (He et al., 2012). All these work are significantly different from what we propose in this work. Also they are conducted in a domain other than Twitter. Most recently, Si et al. (2013) propose a continuous Dirichlet Process Mixture model for Twitter sentiment, for the purpose of stock prediction. Unfortunately there is no evaluation on the accuracy of sentiment classification alone in that work. Furthermore, no standard training or test corpus is used, which makes comparison with other approaches difficult.

Our work is organized in the following way:

- We first propose a universal sentiment model that utilizes various features and resources. The universal model outperforms the top system submitted to the SemEval-2013 task (Mohammad et al., 2013), which was trained and tested on the same data. The universal model serves as a strong baseline and also provides an option for smoothing later.
- We introduce a topic-based mixture model for Twitter sentiment. The model is integrated in the framework of semi-supervised training that takes advantage of large amount of un-annotated Twitter data. Such a mixture model results in further improvement on the sentiment classification accuracy.
- We propose a smoothing technique through interpolation between universal model and topic-based mixture model.
- We also compare different approaches for topic modeling, such as cross-domain topic identification by utilizing data from newswire domain.

2 Universal Sentiment Classifier

In this section we present a universal topic-independent sentiment classifier to establish a state-of-the-art baseline. The sentiment labels are either positive, neutral or negative.

2.1 SVM Classifier

Support Vector Machine (SVM) is an effective classifier that can achieve good performance in high-dimensional feature space. An SVM model represents the examples as points in space, mapped so that the examples of the different categories are separated by a clear margin as wide as possible. In this work an SVM classifier is trained with LibSVM (Chang and Lin, 2011), a widely used toolkit. The linear kernel is found to achieve higher accuracy than other kernels in our initial experiments. The option of probability estimation in LibSVM is turned on so that it can produce the probability of sentiment class c given tweet x at the classification time, i.e. $P(c|x)$.

2.2 Features

The training and testing data are run through tweet-specific tokenization, similar to that used in the CMU Twitter NLP tool (Gimpel et al., 2011).

It is shown in Section 5 that such customized tokenization is helpful. Here are the features that we use for classification:

- Word N-grams: if certain N-gram (unigram, bigram, trigram or 4-gram) appears in the tweet, the corresponding feature is set to 1, otherwise 0. These features are collected from training data, with a count cutoff to avoid overtraining.
- Manual lexicons: it has been shown in other work (Nakov et al., 2013) that lexicons with positive and negative words are important to sentiment classification. In this work, we adopt the lexicon from Bing Liu (Hu and Liu, 2004) which includes about 2000 positive words and 4700 negative words. We also experimented with the popular MPQA (Wilson et al., 2005) lexicon but found no extra improvement on accuracies. A short list of Twitter-specific positive/negative words are also added to enhance the lexicons. We generate two features based on the lexicons: total number of positive words or negative words found in each tweet.
- Emoticons: it is known that people use emoticons in social media data to express their emotions. A set of popular emoticons are collected from the Twitter data we have. Two features are created to represent the presence or absence of any positive/negative emoticons.
- Last sentiment word: a “sentiment word” is any word in the positive/negative lexicons mentioned above. If the last sentiment word found in the tweet is positive (or negative), this feature is set to 1 (or -1). If none of the words in the tweet is sentiment word, it is set to 0 by default.
- PMI unigram lexicons: in (Mohammad et al., 2013) two lexicons were automatically generated based on pointwise mutual information (PMI). One is *NRC Hashtag Sentiment Lexicon* with 54K unigrams, and the other is *Sentiment140 Lexicon* with 62K unigrams. Each word in the lexicon has an associated sentiment score. We compute 7 features based on each of the two lexicons: (1) sum of sentiment score; (2) total number of

positive words (with score $s > 1$); (3) total number of negative words ($s < -1$); (4) maximal positive score; (5) minimal negative score; (6) score of the last positive words; (7) score of the last negative words. Note that for the second and third features, we ignore those with sentiment scores between -1 and 1, since we found that inclusion of those weak subjective words results in unstable performance.

- PMI bigram lexicon: there are also 316K bigrams in the *NRC Hashtag Sentiment Lexicon*. For bigrams, we did not find the sentiment scores useful. Instead, we only compute two features based on counts only: total number of positive bigrams; total number of negative bigrams.
- Punctuations: if there exists exclamation mark or question mark in the tweet, the feature is set to 1, otherwise set to 0.
- Hashtag count: the number of hashtags in each tweet.
- Negation: we collect a list of negation words, including some informal words frequently observed in online conversations, such as “*dunno*” (“don’t know”), “*nvr*” (“never”), etc. For any sentiment words within a window following a negation word and not after punctuations ‘.’, ‘;’, ‘:’, ‘?’, or ‘!’, we reverse its sentiment from positive to negative, or vice versa, before computing the lexicon-based features mentioned earlier. The window size was set to 4 in this work.
- Elongated words: the number of words in the tweet that have letters repeated by at least 3 times in a row, e.g. the word “*goood*”.

3 Topic-Based Sentiment Mixture

3.1 Topic Modeling

Latent Dirichlet Allocation (LDA) (Blei et al., 2003) is one of the widely adopted generative models for topic modeling. The fundamental idea is that a document is a mixture of topics. For each document there is a multinomial distribution over topics, and a Dirichlet prior $Dir(\alpha)$ is introduced on such distribution. For each topic, there is another multinomial distribution over words. One of the popular algorithms for LDA model parameter

estimation and inference is Gibbs sampling (Griffiths and Steyvers, 2004), a form of Markov Chain Monte Carlo. We adopt the efficient implementation of Gibbs sampling as proposed in (Yao et al., 2009) in this work.

Each tweet is regarded as one document. We conduct pre-processing by removing stop words and some of the frequent words found in Twitter data. Suppose that there are T topics in total in the training data, i.e. t_1, t_2, \dots, t_T . The posterior probability of each topic given tweet x_i is computed as in Eq. 1:

$$P_t(t_j|x_i) = \frac{C_{ij} + \alpha_j}{\sum_{k=1}^T C_{ik} + T\alpha_j} \quad (1)$$

where C_{ij} is the number of times that topic t_j is assigned to some word in tweet x_i , usually averaged over multiple iterations of Gibbs sampling. α_j is the j -th dimension of the hyperparameter of Dirichlet distribution that can be optimized during model estimation.

3.2 Sentiment Mixture Model

Once we identify the topics for tweets in the training data, we can split the data into multiple subsets based on topic distributions. For each subset, a separate sentiment model can be trained. There are many ways of splitting the data. For example, K-means clustering can be conducted based on the similarity between the topic distribution vectors or their transformed versions. In this work, we assign tweet x_i to cluster j if $P_t(t_j|x_i) > \tau$ or $P_t(t_j|x_i) = \max_k P_t(t_k|x_i)$. Note that this is a soft clustering, with some tweets possibly assigned to multiple topic-specific clusters. Similar to the universal model, we train T topic-specific sentiment models with LibSVM.

During classification on test tweets, we run topic inference and sentiment classification with multiple sentiment models. They jointly determine the final probability of sentiment class c given tweet x_i as the following in a sentiment mixture model:

$$P(c|x_i) = \sum_{j=1}^T P_m(c|t_j, x_i)P_t(t_j|x_i) \quad (2)$$

where $P_m(c|t_j, x_i)$ is the probability of sentiment c from topic-specific sentiment model trained on topic t_j .

3.3 Smoothing

Additionally, we also experiment with a smoothing technique through linear interpolation between the universal sentiment model and topic-based sentiment mixture model.

$$P(c|x_i) = \theta \times P_U(c|x_i) + (1 - \theta) \times \sum_{j=1}^T P_m(c|t_j, x_i) P_t(t_j|x_i) \quad (3)$$

where θ is the interpolation parameter and $P_U(c|x_i)$ is the probability of sentiment c given tweet x_i from the universal sentiment model.

4 Semi-supervised Training

In this section we propose an integrated framework of semi-supervised training that contains both topic modeling and sentiment classification. The idea of semi-supervised training is to take advantage of large amount low-cost un-annotated data (tweets in this case) to further improve the accuracy of sentiment classification. The algorithm is as follows:

1. Set training corpus D for sentiment classification to be the annotated training data D_a ;
2. Train a sentiment model with current training corpus D ;
3. Run sentiment classification on the un-annotated data D_u with the current sentiment model and generate probabilities of sentiment classes for each tweet, $P(c|x_i)$;
4. Perform data selection. For those tweets with $P(c|x_i) > p$, add them to current training corpus D . The rest is used to replace the un-annotated corpus D_u ;
5. Train a topic model on D , and store the topic inference model and topic distributions of each tweet;
6. Cluster data in D based on the topic distributions from Step 5 and train a separate sentiment model for each cluster. Replace current sentiment model with the new sentiment mixture model;
7. Repeat from Step 3 until finishing a pre-determined number of iterations or no more data is added to D in Step 4.

5 Experimental Results

5.1 Data and Evaluation

We conduct experiments on the data from the task B of *Sentiment Analysis in Twitter* in SemEval-2013. The distribution of positive, neutral and negative data is shown in Table 1. The development set is used to tune parameters and features. The test set is for the blind evaluation.

Set	Pos	Neu	Neg	Total
Training	3640	4586	1458	9684
Dev	575	739	340	1654
Test	1572	1640	601	3813

Table 1: Data from SemEval-2013. Pos: positive; Neu: neutral; Neg: negative.

For semi-supervised training experiments, we explored two sets of additional data. The first one contains 2M tweets randomly sampled from the collection in January and February 2014. The other contains 74K news documents with 50M words collected during the first half year of 2013 from online newswire.

For evaluation, we use macro averaged F score as in (Nakov et al., 2013), i.e. average of the F scores computed on positive and negative classes only. Note that this does not make the task a binary classification problem. Any errors related to neutral class (false positives or false negatives) will negatively impact the F scores.

5.2 Universal Model

In Table 2, we show the incremental improvement in adding various features described in Section 2, measured on the test set. In addition to the features, we also find SVM weighting on the training samples is helpful. Due to the skewness in class distribution in the training set, it is observed during error analysis on the development set that subjective (positive/negative) tweets are more likely to be classified as neutral tweets. The weights for positive, neutral and negative samples are set to be (1, 0.4, 1) based on the results on the development set. As shown in Table 2, weighting adds a 2% improvement. With all features combined, the universal sentiment model achieves 69.7 on average F score. The F score from the best system in SemEval-2013 (Mohammad et al., 2013) is also listed in the last row of Table 2 for a comparison.

Model	Avg. F score
Baseline with word N-grams	55.0
+ tweet tokenization	56.1
+ manual lexicon features	62.4
+ emoticons	62.8
+ last sentiment word	63.7
+ PMI unigram lexicons	64.5
+ hashtag counts	65.0
+ SVM weighting	67.0
+ PMI bigram lexicons	68.2
+ negations	69.0
+ elongated words	69.7
Mohammad et al., 2013	69.0

Table 2: Results on the test set with universal sentiment model.

5.3 Topic-Based Mixture Model

For the topic-based mixture model and semi-supervised training, based on the experiments on the development set, we set the parameter τ used in soft clustering to 0.4, the data selection parameter p to 0.96, and the interpolation parameter for smoothing θ to 0.3. We found no more noticeable benefits after two iterations of semi-supervised training. The number of topics is set to 100.

The results on the test set are shown Table 3, with the topic information inferred from either Twitter data (second column) or newswire data (third column). The first row shows the performance of the universal sentiment model as a baseline. The second row shows the results from re-training the universal model by simply adding tweets selected from two iterations of semi-supervised training (about 100K). It serves as another baseline with more training data, for a fair comparison with the topic-based mixture modeling that uses the same amount of training data.

We also conduct an experiment by only considering the most likely topic for each tweet when computing the sentiment probabilities. The results show that the topic-based mixture model outperforms both the baseline and the one that considers the top topics only. Smoothing with the universal model adds further improvement in addition to the un-smoothed mixture model. With the topic information inferred from Twitter data, the F score is 2 points higher than the baseline without semi-

Model	Tweet-topic	News-topic
Baseline	69.7	69.7
+ semi-supervised	70.3	70.2
top topic only	70.6	70.4
mixture	71.2	70.8
+ smoothing	71.7	71.1

Table 3: Results of topic-based sentiment mixture model on SemEval test set.

supervised training and 1.4 higher than the baseline with semi-supervised data.

As shown in the third column in Table 3, surprisingly, the model with topic information inferred from the newswire data works well on the Twitter domain. A 1.4 points of improvement can be obtained compared to the baseline. This provides an opportunity for cross-domain topic identification when data from certain domain is more difficult to obtain than others.

In Table 4, we provide some examples from the topics identified in tweets as well as the newswire data. The most frequent words in each topic are listed in the table. We can clearly see that the topics are about phones, sports, sales and politics, respectively.

Tweet-1	Tweet-2	News-1	News-2
phone	game	sales	party
call	great	stores	government
answer	play	online	election
question	team	retail	minister
service	win	store	political
text	tonight	retailer	prime
texting	super	business	state

Table 4: The most frequent words in example topics from tweets and newswire data.

6 Conclusions

In this paper, we presented multiple approaches for advanced Twitter sentiment analysis. We established a state-of-the-art baseline that utilizes a variety of features, and built a topic-based sentiment mixture model with topic-specific Twitter data, all integrated in a semi-supervised training framework. The proposed model outperforms the top system in SemEval-2013. Further research is needed to continue to improve the accuracy in this difficult domain.

References

- David Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *In Journal of Machine Learning Research*. 3(2003), 993–1022.
- S. R. K. Branavan, Harr Chen, Jacob Eisenstein, and Regina Barzilay. 2008. Learning document-level semantic properties from free-text annotations. *In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2008)*.
- Chih-Chung Chang and Chih-Jen Lin. 2011. LIB-SVM: A library for support vector machines. *In ACM Transactions on Intelligent Systems and Technology*.
- Kevin Gimpel, Nathan Schneider, Brendan O’Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for Twitter: annotation, features, and experiments. *In Proceedings of the Annual Meeting of the Association for Computational Linguistics*.
- Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *In Proceedings of the National Academy of Science*. 101, 5228–5235.
- Yulan He, Chenghua Lin, Wei Gao, and Kam-Fai Wong. 2012. Tracking sentiment and topic dynamics from social media. *In Proceedings of the 6th International AAAI Conference on Weblogs and Social Media (ICWSM-2012)*.
- Mingqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. *In Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
- Yohan Jo and Alice Oh. 2011. Aspect and sentiment unification model for online review analysis. *In Proceedings of ACM Conference in Web Search and Data Mining (WSDM-2011)*.
- Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. *In Proceedings of International Conference on World Wide Web (WWW-2007)*.
- Saif M. Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. *In Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. 312-320, Atlanta, Georgia, June 14-15, 2013.
- Preslav Nakov, Zornitsa Kozareva, Alan Ritter, Sara Rosenthal, Veselin Stoyanov, and Theresa Wilson. 2013. SemEval-2013 Task 2: Sentiment Analysis in Twitter. *In Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. 312-320, Atlanta, Georgia, June 14-15, 2013.
- Jianfeng Si, Arjun Mukherjee, Bing Liu, Qing Li, Huayi Li, and Xiaotie Deng. 2013. Exploiting topic based Twitter sentiment for stock prediction. *In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. 24-29, Sofia, Bulgaria, August 4-9, 2013.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. *In Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT 05*.
- Limin Yao, David Mimno, and Andrew McCallum. 2009. Efficient methods for topic model inference on streaming document collections. *KDD’09*.

Cross-cultural Deception Detection

Verónica Pérez-Rosas

Computer Science and Engineering
University of North Texas
veronicaperezrosas@my.unt.edu

Rada Mihalcea

Computer Science and Engineering
University of Michigan
mihalcea@umich.edu

Abstract

In this paper, we address the task of cross-cultural deception detection. Using crowdsourcing, we collect three deception datasets, two in English (one originating from United States and one from India), and one in Spanish obtained from speakers from Mexico. We run comparative experiments to evaluate the accuracies of deception classifiers built for each culture, and also to analyze classification differences within and across cultures. Our results show that we can leverage cross-cultural information, either through translation or equivalent semantic categories, and build deception classifiers with a performance ranging between 60-70%.

1 Introduction

The identification of deceptive behavior is a task that has gained increasing interest from researchers in computational linguistics. This is mainly motivated by the rapid growth of deception in written sources, and in particular in Web content, including product reviews, online dating profiles, and social networks posts (Ott et al., 2011).

To date, most of the work presented on deception detection has focused on the identification of deceit clues within a specific language, where English is the most commonly studied language. However, a large portion of the written communication (e.g., e-mail, chats, forums, blogs, social networks) occurs not only between speakers of English, but also between speakers from other cultural backgrounds, which poses important questions regarding the applicability of existing deception tools. Issues such as language, beliefs, and moral values may influence the way people deceive, and therefore may have implications on the construction of tools for deception detection.

In this paper, we explore within- and across-culture deception detection for three different cultures, namely United States, India, and Mexico. Through several experiments, we compare the performance of classifiers that are built separately for each culture, and classifiers that are applied across cultures, by using unigrams and word categories that can act as a cross-lingual bridge. Our results show that we can achieve accuracies in the range of 60-70%, and that we can leverage resources available in one language to build deception tools for another language.

2 Related Work

Research to date on automatic deceit detection has explored a wide range of applications such as the identification of spam in e-mail communication, the detection of deceitful opinions in review websites, and the identification of deceptive behavior in computer-mediated communication including chats, blogs, forums and online dating sites (Peng et al., 2011; Toma et al., 2008; Ott et al., 2011; Toma and Hancock, 2010; Zhou and Shi, 2008).

Techniques used for deception detection frequently include word-based stylometric analysis. Linguistic clues such as n-grams, count of used words and sentences, word diversity, and self-references are also commonly used to identify deception markers. An important resource that has been used to represent semantic information for the deception task is the Linguistic Inquiry and Word Count (LIWC) dictionary (Pennebaker and Francis, 1999). LIWC provides words grouped into semantic categories relevant to psychological processes, which have been used successfully to perform linguistic profiling of true tellers and liars (Zhou et al., 2003; Newman et al., 2003; Rubin, 2010). In addition to this, features derived from syntactic Context Free Grammar parse trees, and part of speech have also been found to aid the deceit detection (Feng et al., 2012; Xu and Zhao, 2012).

While most of the studies have focused on English, there is a growing interest in studying deception for other languages. For instance, (Fornaciari and Poesio, 2013) identified deception in Italian by analyzing court cases. The authors explored several strategies for identifying deceptive clues, such as utterance length, LIWC features, lemmas and part of speech patterns. (Almela et al., 2012) studied the deception detection in Spanish text by using SVM classifiers and linguistic categories, obtained from the Spanish version of the LIWC dictionary. A study on Chinese deception is presented in (Zhang et al., 2009), where the authors built a deceptive dataset using Internet news and performed machine learning experiments using a bag-of-words representation to train a classifier able to discriminate between deceptive and truthful cases.

It is also worth mentioning the work conducted to analyze cross-cultural differences. (Lewis and George, 2008) presented a study of deception in social networks sites and face-to-face communication, where authors compare deceptive behavior of Korean and American participants, with a subsequent study also considering the differences between Spanish and American participants (Lewis and George, 2009). In general, research findings suggest a strong relation between deception and cultural aspects, which are worth exploring with automatic methods.

3 Datasets

We collect three datasets for three different cultures: United States (English-US), India (English-India), and Mexico (Spanish-Mexico). Following (Mihalcea and Strapparava, 2009), we collect short deceptive and truthful essays for three topics: opinions on Abortion, opinions on Death Penalty, and feelings about a Best Friend.

For English-US and English-India, we use Amazon Mechanical Turk with a location restriction, so that all the contributors are from the country of interest (US and India). We collect 100 deceptive and 100 truthful statements for each of the three topics. To avoid spam, each contribution is manually verified by one of the authors of this paper. For Spanish-Mexico, while we initially attempted to collect data also using Mechanical Turk, we were not able to receive enough contributions. We therefore created a separate web interface to collect data, and recruited participants through contacts of the paper's authors. The overall process was significantly more time consuming than for the other two cul-

tures, and resulted in fewer contributions, namely 39+39 statements for Abortion, 42+42 statements for Death Penalty, and 94+94 statements for Best Friend. For all three cultures, the participants first provided their truthful responses, followed by the deceptive ones.

Interestingly, for all three cultures, the average number of words for the deceptive statements (62 words) is significantly smaller than for the truthful statements (81 words), which may be explained by the added difficulty of the deceptive process, and is in line with previous observations about the cues of deception (DePaulo et al., 2003).

4 Experiments

Through our experiments, we seek answers to the following questions. First, what is the performance for deception classifiers built for different cultures? Second, can we use information drawn from one culture to build a deception classifier for another culture? Finally, what are the psycholinguistic classes most strongly associated with deception/truth, and are there commonalities or differences among languages?

In all our experiments, we formulate the deception detection task in a machine learning framework, where we use an SVM classifier to discriminate between deceptive and truthful statements.¹

4.1 What is the performance for deception classifiers built for different cultures?

We represent the deceptive and truthful statements using two different sets of features. First we use unigrams obtained from the statements corresponding to each topic and each culture. To select the unigrams, we use a threshold of 10, where all the unigrams with a frequency less than 10 are dropped. Since previous research suggested that stopwords can contain linguistic clues for deception, no stopword removal is performed.

Experiments are performed using a ten-fold cross validation evaluation on each dataset. Using the same unigram features, we also perform cross-topic classification, so that we can better understand the topic dependence. For this, we train the SVM classifier on training data consisting of a merge of two topics (e.g., Abortion + Best Friend) and test on the third topic (e.g., Death Penalty). The results for both within- and cross-topic are shown in the last two columns of Table 1.

¹We use the SVM classifier implemented in the Weka toolkit, with its default settings.

Topic	LIWC				Unigrams		
	Linguistic	Psychological	Relativity	Personal	All	Within-topic	Cross-topic
English-US							
Abortion	72.50%	68.75%	44.37%	67.50%	73.03%	63.75%	80.36%
Best Friend	75.98%	68.62%	58.33%	54.41%	73.03%	74.50%	60.78%
Death Penalty	60.36%	54.50%	49.54%	50.45%	58.10%	58.10%	77.23%
Average	69.61%	63.96%	50.75%	57.45%	69.05%	65.45%	72.79%
English-India							
Abortion	56.00%	48.50%	46.50%	48.50%	56.00%	46.00%	50.00%
Best Friend	68.18%	68.62%	54.55%	53.18%	71.36%	60.45%	57.23%
Death Penalty	56.00%	52.84%	57.50%	53.50%	63.50%	57.50%	54.00%
Average	60.06%	59.19%	52.84%	51.72%	63.62%	54.65%	53.74%
Spanish-Mexico							
Abortion	73.17%	67.07%	48.78%	51.22%	62.20%	52.46%	57.69%
Best Friend	72.04%	74.19%	67.20%	54.30%	75.27%	66.66%	50.53%
Death Penalty	73.17%	67.07%	48.78%	51.22%	62.20%	54.87%	63.41%
Average	72.79%	69.45%	54.92%	52.25%	67.89%	57.99%	57.21%

Table 1: Within-culture classification, using LIWC word classes and unigrams. For LIWC, results are shown for within-topic experiments, with ten-fold cross validation. For unigrams, both within-topic (ten-fold cross validation on the same topic) and cross-topic (training on two topics and testing on the third topic) results are reported.

Second, we use the LIWC lexicon to extract features corresponding to several word classes. LIWC was developed as a resource for psycholinguistic analysis (Pennebaker and Francis, 1999). The 2001 version of LIWC includes about 2,200 words and word stems grouped into about 70 classes relevant to psychological processes (e.g., emotion, cognition), which in turn are grouped into four broad categories² namely: linguistic processes, psychological processes, relativity, and personal concerns. A feature is generated for each of the 70 word classes by counting the total frequency of the words belonging to that class. We perform separate evaluations using each of the four broad LIWC categories, as well as using all the categories together. The results obtained with the SVM classifier are shown in Table 1.

Overall, the results show that it is possible to discriminate between deceptive and truthful cases using machine learning classifiers, with a performance superior to a random baseline which for all datasets is 50% given an even class distribution. Considering the unigram results, among the three cultures considered, the deception discrimination works best for the English-US dataset, and this is also the dataset that benefits most from the larger amount of training data brought by the cross-topic experiments. In general, the cross-topic evaluations suggest that there is no high topic dependence in this task, and that using deception data from differ-

ent topics can lead to results that are comparable to the within-topic data. Interestingly, among the three topics considered, the Best Friend topic has consistently the highest within-topic performance, which may be explained by the more personal nature of the topic, which can lead to clues that are useful for the detection of deception (e.g., references to the self or personal relationships).

Regarding the LIWC classifiers, the results show that the use of the LIWC classes can lead to performance that is generally better than the one obtained with the unigram classifiers. The explicit categorization of words into psycholinguistic classes seems to be particularly useful for the languages where the words by themselves did not lead to very good classification accuracies. Among the four broad LIWC categories, the linguistic category appears to lead to the best performance as compared to the other categories. It is notable that in Spanish, the linguistic category by itself provides results that are better than when all the LIWC classes are used, which may be due to the fact that Spanish has more explicit lexicalization for clues that may be relevant to deception (e.g., verb tenses, formality).

4.2 Can we use information drawn from one culture to build a deception classifier in another culture?

In the next set of experiments, we explore the detection of deception using training data originating from a different culture. As with the within-culture

²<http://www.liwc.net/descriptiontable1.php>

Topic	Linguistic	Psychological	Relativity	Personal	All LIWC	Unigrams
Training: English-US Test: English-India						
Abortion	58.00%	51.00%	48.50%	51.50%	52.25%	57.89%
Best Friend	66.36%	47.27%	48.64%	50.45%	59.54%	51.00%
Death Penalty	54.50%	50.50%	50.00%	48.50%	53.5%	59.00%
Average	59.62%	49.59%	49.05%	50.15%	55.10%	55.96%
Training: English-India Test: English-US						
Abortion	71.32%	47.49%	43.38%	45.82%	62.50%	55.51%
Best Friend	59.74%	49.35%	51.94%	49.36%	55.84%	53.20%
Death Penalty	51.47%	44.11%	54.88%	50.98%	39.21%	50.71%
Average	60.87%	46.65%	50.06%	48.72%	52.51%	54.14%
Training: English-US Test: Spanish-Mexico						
Abortion	70.51%	46.15%	50.00%	52.56%	53.85%	61.53%
Best Friend	69.35%	52.69%	51.08%	46.77%	67.74%	65.03%
Death Penalty	54.88%	54.88%	53.66%	50.00%	62.19%	59.75%
Average	64.92%	51.24%	51.58%	49.78%	61.26%	62.10%
Training: English-India Test: Spanish-Mexico						
Abortion	48.72%	50.00%	47.44%	42.31%	43.58%	55.12%
Best Friend	68.28%	63.44%	56.45%	54.84%	60.75%	67.20%
Death Penalty	60.98%	53.66%	54.88%	60.98%	59.75%	51.21%
Average	59.32%	55.70%	52.92%	52.71%	54.69%	57.84%

Table 2: Cross-cultural experiments using LIWC categories and unigrams

experiments, we use unigrams and LIWC features. For consistency across the experiments, given that the size of the Spanish dataset is different compared to the other two datasets, we always train on one of the English datasets.

To enable the unigram based experiments, we translate the two English datasets into Spanish by using the Bing API for automatic translation.³ As before, we extract and keep only the unigrams with frequency greater or equal to 10. The results obtained in these cross-cultural experiments are shown in the last column of Table 2.

In a second set of experiments, we use the LIWC word classes as a bridge between languages. First, each deceptive or truthful statement is represented using features based on the LIWC word classes. Next, since the same word classes are used in both the English and the Spanish LIWC lexicons, this LIWC-based representation is independent of language, and therefore can be used to perform cross-cultural experiments. Table 2 shows the results obtained with each of the four broad LIWC categories, as well as with all the LIWC word classes.

We also attempted to combine unigrams and LIWC features. However, in most cases, no improvements were noticed with respect to the use of unigrams or LIWC features alone. We are not reporting these results due to space limitation.

These cross-cultural evaluations lead to several

findings. First, we can use data from a culture to build deception classifiers for another culture, with performance figures better than the random baseline, but weaker than the results obtained with within-culture data. An important finding is that LIWC can be effectively used as a bridge for cross-cultural classification, with results that are comparable to the use of unigrams, which suggests that such specialized lexicons can be used for cross-cultural or cross-lingual classification. Moreover, using only the linguistic category from LIWC brings additional improvements, with absolute improvements of 2-4% over the use of unigrams. This is an encouraging result, as it implies that a semantic bridge such as LIWC can be effectively used to classify deception data in other languages, instead of using the more costly and time consuming unigram method based on translations.

4.3 What are the psycholinguistic classes most strongly associated with deception/truth?

The final question we address is concerned with the LIWC classes that are dominant in deceptive and truthful text for different cultures. We use the method presented in (Mihalcea and Strapparava, 2009), which consists of a metric that measures the saliency of LIWC classes in deceptive versus truthful data. Following their strategy, we first create a corpus of deceptive and truthful text using a mix of all the topics in each culture. We then calculate

³<http://http://http://www.bing.com/dev/en-us/dev-center>

Class	Score	Sample words	Class	Score	Sample words
English-US					
Deceptive			Truthful		
Metaph	1.77	Die,died,hell,sin,lord	Insight	0.68	Accept,believe,understand
Other	1.46	He,her,herself,him	I	0.66	I,me,my,myself,
You	1.41	Thou,you	Optimism	0.65	accept, hope, top, best
Othref	1.18	He,her,herself,him	We	0.55	Our,ourselves,us,we,
Negemo	1.18	Afraid,agony,awful,bad	Friends	0.46	Buddies,friend
English-India					
Deceptive			Truthful		
Negate	1.49	Cannot,neither,no,none	Past	0.78	Happened,helped,liked,listened
Physical	1.46	Heart,ill,love,loved,	I	0.66	I,me,mine,my
Future	1.42	Be,may,might,will	Optimism	0.65	Accept,accepts,best,bold,
Other	1.17	He,she, himself,herself	We	0.55	Our,ourselves,us,we
Humans	1.08	Adult,baby,children,human	Friends	0.46	Buddies,companion,friend,pal
Spanish-Mexico					
Deceptive			Truthful		
Certain	1.47	Jamás(never),siempre(always)	Optimism	0.66	Aceptar(accept),animar(cheer)
Humans	1.28	Bebé(baby),persona(person)	Self	0.65	Conmigo(me),tengo(have),soy(am)
You	1.26	Eres(are),estas(be),su(his/her)	We	0.58	Estamos(are),somos(be),tenemos(have)
Negate	1.25	Jamás(never),tampoco(neither)	Friends	0.37	Amigo/amiga(friend),amistad(friendship)
Other	1.22	Es(is),esta(are),otro(other)	Past	0.32	Compartimos(share),vivimos(lived)

Table 3: Top ranked LIWC classes for each culture, along with sample words

the dominance for each LIWC class, and rank the classes in reversed order of their dominance score. Table 3 shows the most salient classes for each culture, along with sample words.

This analysis shows some interesting patterns. There are several classes that are shared among the cultures. For instance, the deceivers in all cultures make use of negation, negative emotions, and references to others. Second, true tellers use more optimism and friendship words, as well as references to themselves. These results are in line with previous research, which showed that LIWC word classes exhibit similar trends when distinguishing between deceptive and non-deceptive text (Newman et al., 2003). Moreover, there are also word classes that only appear in some of the cultures; for example, time classes (Past, Future) appear in English-India and Spanish-Mexico, but not in English-US, which in turn contains other classes such as Insight and Metaph.

5 Conclusions

In this paper, we addressed the task of deception detection within- and across-cultures. Using three datasets from three different cultures, each covering three different topics, we conducted several experiments to evaluate the accuracy of deception detection when learning from data from the same culture or from a different culture. In our evaluations, we compared the use of unigrams versus the

use of psycholinguistic word classes.

The main findings from these experiments are: 1) We can build deception classifiers for different cultures with accuracies ranging between 60-70%, with better performance obtained when using psycholinguistic word classes as compared to simple unigrams; 2) The deception classifiers are not sensitive to different topics, with cross-topic classification experiments leading to results comparable to the within-topic experiments; 3) We can use data originating from one culture to train deception detection classifiers for another culture; the use of psycholinguistic classes as a bridge across languages can be as effective or even more effective than the use of translated unigrams, with the added benefit of making the classification process less costly and less time consuming.

The datasets introduced in this paper are publicly available from <http://nlp.eecs.umich.edu>.

Acknowledgments

This material is based in part upon work supported by National Science Foundation awards #1344257 and #1355633 and by DARPA-BAA-12-47 DEFT grant #12475008. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or the Defense Advanced Research Projects Agency.

References

- Á. Almela, R. Valencia-García, and P. Cantos. 2012. Seeing through deception: A computational approach to deceit detection in written communication. In *Proceedings of the Workshop on Computational Approaches to Deception Detection*, pages 15–22, Avignon, France, April. Association for Computational Linguistics.
- B. DePaulo, J. Lindsay, B. Malone, L. Muhlenbruck, K. Charlton, and H. Cooper. 2003. Cues to deception. *Psychological Bulletin*, 129(1).
- S. Feng, R. Banerjee, and Y. Choi. 2012. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pages 171–175, Stroudsburg, PA, USA. Association for Computational Linguistics.
- T. Fornaciari and M. Poesio. 2013. Automatic deception detection in italian court cases. *Artificial Intelligence and Law*, 21(3):303–340.
- C. Lewis and J. George. 2008. Cross-cultural deception in social networking sites and face-to-face communication. *Comput. Hum. Behav.*, 24(6):2945–2964, September.
- C. Lewis and Giordano G. George, J. 2009. A cross-cultural comparison of computer-mediated deceptive communication. In *Proceedings of Pacific Asia Conference on Information Systems*.
- R. Mihalcea and C. Strapparava. 2009. The lie detector: Explorations in the automatic recognition of deceptive language. In *Proceedings of the Association for Computational Linguistics (ACL 2009)*, Singapore.
- M. Newman, J. Pennebaker, D. Berry, and J. Richards. 2003. Lying words: Predicting deception from linguistic styles. *Personality and Social Psychology Bulletin*, 29.
- M. Ott, Y. Choi, C. Cardie, and J. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 309–319, Stroudsburg, PA, USA. Association for Computational Linguistics.
- H. Peng, C. Xiaoling, C. Na, R. Chandramouli, and P. Subbalakshmi. 2011. Adaptive context modeling for deception detection in emails. In *Proceedings of the 7th international conference on Machine learning and data mining in pattern recognition*, MLDM'11, pages 458–468, Berlin, Heidelberg. Springer-Verlag.
- J. Pennebaker and M. Francis. 1999. Linguistic inquiry and word count: LIWC. Erlbaum Publishers.
- V. Rubin. 2010. On deception and deception detection: Content analysis of computer-mediated stated beliefs. *Proceedings of the American Society for Information Science and Technology*, 47(1):1–10.
- C. Toma and J. Hancock. 2010. Reading between the lines: linguistic cues to deception in online dating profiles. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work, CSCW '10*, pages 5–8, New York, NY, USA. ACM.
- C. Toma, J. Hancock, and N. Ellison. 2008. Separating fact from fiction: An examination of deceptive self-presentation in online dating profiles. *Personality and Social Psychology Bulletin*, 34(8):1023–1036.
- Q. Xu and H. Zhao. 2012. Using deep linguistic features for finding deceptive opinion spam. In *Proceedings of COLING 2012: Posters*, pages 1341–1350, Mumbai, India, December. The COLING 2012 Organizing Committee.
- H. Zhang, S. Wei, H. Tan, and J. Zheng. 2009. Deception detection based on svm for chinese text in cmc. In *Information Technology: New Generations, 2009. ITNG '09. Sixth International Conference on*, pages 481–486, April.
- L. Zhou and D. Shi, Y. and Zhang. 2008. A statistical language modeling approach to online deception detection. *IEEE Trans. on Knowl. and Data Eng.*, 20(8):1077–1081, August.
- L Zhou, D. Twitchell, T Qin, J. Burgoon, and J. Nuna-maker. 2003. An exploratory study into deception detection in text-based computer-mediated communication. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03) - Track1 - Volume 1*, HICSS '03, pages 44.2–, Washington, DC, USA. IEEE Computer Society.

Particle Filter Rejuvenation and Latent Dirichlet Allocation

Chandler May,[†] Alex Clemmer[‡] and Benjamin Van Durme[†]

[†]Human Language Technology Center of Excellence

Johns Hopkins University

[‡]Microsoft

cjmay@jhu.edu, clemmer.alexander@gmail.com, vandurme@cs.jhu.edu

Abstract

Previous research has established several methods of *online* learning for latent Dirichlet allocation (LDA). However, *streaming* learning for LDA—allowing only one pass over the data and constant storage complexity—is not as well explored. We use reservoir sampling to reduce the storage complexity of a previously-studied online algorithm, namely the particle filter, to constant. We then show that a simpler particle filter implementation performs just as well, and that the quality of the initialization dominates other factors of performance.

1 Introduction

We extend a popular model, latent Dirichlet allocation (LDA), to unbounded streams of documents. In order for inference to be practical in this setting it must use constant space asymptotically and run in pseudo-linear time, perhaps $O(n)$ or $O(n \log n)$.

Canini et al. (2009) presented a method for LDA inference based on particle filters, where a sample set of models is updated online with each new token observed from a stream. In general, these models should be regularly resampled and rejuvenated using Markov Chain Monte Carlo (MCMC) steps over the history in order to improve the efficiency of the particle filter (Gilks and Berzuini, 2001). The particle filter of Canini et al. (2009) rejuvenates over independent draws from the history by storing all past observations and states. This algorithm thus has linear storage complexity and is not an online learning algorithm in a strict sense (Börschinger and Johnson, 2012).

In the current work we propose using reservoir sampling in the rejuvenation step to reduce the storage complexity of the particle filter to $O(1)$. This improvement is practically useful in the large-data setting and is also scientifically interesting in that it recovers some of the cognitive plausibility which originally motivated Börschinger and Johnson (2012). However, in experiments on the dataset studied by Canini et al. (2009), we show that rejuvenation does not benefit the particle filter’s performance. Rather, performance is dominated by the effects of random initialization (a problem for which we provide a correction while abiding by the same constraints as Canini et al. (2009)). This result re-opens the question of whether rejuvenation is of practical importance in online learning for static Bayesian models.

2 Latent Dirichlet Allocation

For a sequence of N words collected into documents of varying length, we denote the j -th word as w_j , and the document it occurs in as d_i . LDA (Blei et al., 2003) “explains” the occurrence of each word by postulating that a document was generated by repeatedly: (1) sampling a topic z from $\theta^{(d)}$, the document-specific mixture of T topics, and (2) sampling a word w from $\phi^{(z)}$, the probability distribution the z -th topic defines over the vocabulary.

The goal is to infer θ and ϕ , under the model:

$$\begin{aligned}w_i \mid z_i, \phi^{(z_i)} &\sim \text{Categorical}(\phi^{(z_i)}) \\ \phi^{(z)} &\sim \text{Dirichlet}(\beta) \\ z_i \mid \theta^{(d_i)} &\sim \text{Categorical}(\theta^{(d_i)}) \\ \theta^{(d)} &\sim \text{Dirichlet}(\alpha)\end{aligned}$$

```

initialize weights  $\omega_0^{(p)} = 1/P$  for  $p = 1, \dots, P$ 
for  $i = 1, \dots, N$  do
  for  $p = 1, \dots, P$  do
    set  $\omega_i^{(p)} = \omega_{i-1}^{(p)} \mathbf{P}(w_i | \mathbf{z}_{i-1}^{(p)}, \mathbf{w}_{i-1})$ 
    sample  $z_i^{(p)}$  w.p.  $\mathbf{P}(z_i^{(p)} | \mathbf{z}_{i-1}^{(p)}, \mathbf{w}_i)$ .
  if  $\|\omega\|_2^{-2} \leq ESS$  then
    for  $j \in \mathcal{R}(i)$  do
      for  $p = 1, \dots, P$  do
        sample  $z_j^{(p)}$  w.p.
           $\mathbf{P}(z_j^{(p)} | \mathbf{z}_{i \setminus j}^{(p)}, \mathbf{w}_i)$ 
        set  $\omega_i^{(p)} = 1/P$  for each particle

```

Algorithm 1: Particle filtering for LDA.

Computing ϕ and θ exactly is generally intractable, motivating methods for approximate inference such as variational Bayesian inference (Blei et al., 2003), expectation propagation (Minka and Lafferty, 2002), and collapsed Gibbs sampling (Griffiths and Steyvers, 2004).

A limitation of these techniques is they require multiple passes over the data to obtain good samples of ϕ and θ . This requirement makes them impractical when the corpus is too large to fit directly into memory and in particular when the corpus grows without bound. This motivates online learning techniques, including sampling-based methods (Banerjee and Basu, 2007; Canini et al., 2009) and stochastic variational inference (Hoffman et al., 2010; Mimno et al., 2012; Hoffman et al., 2013). However, where these approaches generally assume the ability to draw independent samples from the full dataset, we consider the case when it is infeasible to access arbitrary elements from the history. The one existing algorithm that can be directly applied under this constraint, to our knowledge, is the streaming variational Bayes framework (Broderick et al., 2013) in which the posterior is recursively updated as new data arrives using a variational approximation.

3 Online LDA Using Particle Filters

Particle filters are a family of sequential Monte Carlo (SMC) sampling algorithms designed to estimate the posterior distribution of a system with dynamic state (Doucet et al., 2001). A particle filter approximates the posterior by a weighted sample of points, or particles, from the state space. The particle cloud is updated recursively for each new observation using importance sampling (an approach called *sequential importance sampling*).

Canini et al. (2009) apply this approach to LDA after analytically integrating out ϕ and θ , obtaining a Rao-Blackwellized particle filter (Doucet et al., 2000) that estimates the collapsed posterior $\mathbf{P}(\mathbf{z} | \mathbf{w})$. In this setting, the P particles are samples of the topic assignment vector $\mathbf{z}^{(p)}$, and they are propagated forward in state space one token at a time. In general, the larger P is, the more accurately we approximate the posterior; for small P , the approximation of the tails of the posterior will be particularly poor (Pitt and Shephard, 1999). However, a larger value of P increases the runtime and storage requirements of the algorithm.

We now describe the Rao-Blackwellized particle filter for LDA in detail (pseudocode is given in Algorithm 1). At the moment token i is observed, the particles form a discrete approximation of the posterior up to the $(i-1)$ -th word:

$$\mathbf{P}(\mathbf{z}_{i-1} | \mathbf{w}_{i-1}) \approx \sum_p \omega_{i-1}^{(p)} I_{\mathbf{z}_{i-1}}(\mathbf{z}_{i-1}^{(p)})$$

where $I_{\mathbf{z}}(\mathbf{z}')$ is the indicator function, evaluating to 1 if $\mathbf{z} = \mathbf{z}'$ and 0 otherwise. Now each particle p is propagated forward by drawing a topic $z_i^{(p)}$ from the conditional posterior distribution $\mathbf{P}(z_i^{(p)} | \mathbf{z}_{i-1}^{(p)}, \mathbf{w}_i)$ and scaling the particle weight by $\mathbf{P}(w_i | \mathbf{z}_{i-1}^{(p)}, \mathbf{w}_{i-1})$. The particle cloud now approximates the posterior up to the i -th word:

$$\mathbf{P}(\mathbf{z}_i | \mathbf{w}_i) \approx \sum_p \omega_i^{(p)} I_{\mathbf{z}_i}(\mathbf{z}_i^{(p)}).$$

Dropping the superscript (p) for notational convenience, the conditional posterior used in the propagation step is given by

$$\begin{aligned} \mathbf{P}(z_i | \mathbf{z}_{i-1}, \mathbf{w}_i) &\propto \mathbf{P}(z_i, w_i | \mathbf{z}_{i-1}, \mathbf{w}_{i-1}) \\ &= \frac{n_{z_i, i \setminus i}^{(w_i)} + \beta}{n_{z_i, i \setminus i}^{(\cdot)} + W\beta} \frac{n_{z_i, i \setminus i}^{(d_i)} + \alpha}{n_{\cdot, i \setminus i}^{(d_i)} + T\alpha} \end{aligned}$$

where $n_{z_i, i \setminus i}^{(w_i)}$ is the number of times word w_i has been assigned topic z_i so far, $n_{z_i, i \setminus i}^{(\cdot)}$ is the number of times any word has been assigned topic z_i , $n_{z_i, i \setminus i}^{(d_i)}$ is the number of times topic z_i has been assigned to any word in document d_i , and $n_{\cdot, i \setminus i}^{(d_i)}$ is the number of words observed in document d_i . The particle weights are scaled as

$$\begin{aligned} \frac{\omega_i^{(p)}}{\omega_{i-1}^{(p)}} &\propto \frac{\mathbf{P}(w_i | \mathbf{z}_i^{(p)}, \mathbf{w}_i) \mathbf{P}(z_i^{(p)} | \mathbf{z}_{i-1}^{(p)})}{Q(z_i^{(p)} | \mathbf{z}_{i-1}^{(p)}, \mathbf{w}_i)} \\ &= \mathbf{P}(w_i | \mathbf{z}_{i-1}^{(p)}, \mathbf{w}_{i-1}) \end{aligned}$$

where Q is the proposal distribution for the particle state transition; in our case,

$$Q(z_i^{(p)} | \mathbf{z}_{i-1}^{(p)}, \mathbf{w}_i) = \mathbf{P}(z_i^{(p)} | \mathbf{z}_{i-1}^{(p)}, \mathbf{w}_i),$$

minimizing the variance of the importance weights conditioned on \mathbf{w}_i and \mathbf{z}_{i-1} (Doucet et al., 2000).

Over time the particle weights tend to diverge. To combat this inefficiency, after every state transition we estimate the effective sample size (ESS) of the particle weights as $\|\omega_i\|_2^{-2}$ (Liu and Chen, 1998) and resample the particles when that estimate drops below a prespecified threshold. Several resampling strategies have been proposed (Doucet et al., 2000); we perform multinomial resampling as in Pitt and Shephard (1999) and Ahmed et al. (2011), treating the weights as unnormalized probability masses on the particles.

After resampling we are likely to have several copies of the same particle, yielding a degenerate approximation to the posterior. To reintroduce diversity to the particle cloud we take MCMC steps over a sequence of states from the history (Doucet et al., 2000; Gilks and Berzuini, 2001). We call the indices of these states the rejuvenation sequence, denoted $\mathcal{R}(i)$ (Canini et al., 2009). The transition probability for a state $j \in \mathcal{R}(i)$ is given by

$$\mathbf{P}(z_j | \mathbf{z}_{N \setminus j}, \mathbf{w}_N) \propto \frac{n_{z_j, N \setminus j}^{(w_j)} + \beta}{n_{z_j, N \setminus j}^{(\cdot)} + W\beta} \frac{n_{z_j, N \setminus j}^{(d_j)} + \alpha}{n_{z_j, N \setminus j}^{(d_j)} + T\alpha}$$

where subscript $N \setminus j$ denotes counts up to token N , excluding those for token j .

The rejuvenation sequence can be chosen by the practitioner. Choosing a long sequence (large $|\mathcal{R}(i)|$) may result in a more accurate posterior approximation but also increases runtime and storage requirements. The tokens in $\mathcal{R}(i)$ may be chosen uniformly at random from the history or under a biased scheme that favors recent observations. The particle filter studied empirically by Canini et al. (2009) stored the entire history, incurring linear storage complexity in the size of the stream. Ahmed et al. (2011) instead sampled ten documents from the most recent 1000, achieving constant storage complexity at the cost of a recency bias. If we want to fit a model to a long non-i.i.d. stream, we require an unbiased rejuvenation sequence as well as sub-linear storage complexity.

4 Reservoir Sampling

Reservoir sampling is a widely-used family of algorithms for choosing an array (“reservoir”) of k

items. The most common example, presented in Vitter (1985) as Algorithm R, chooses k elements of a stream such that each possible subset of k elements is equiprobable. This effects sampling k items uniformly without replacement, using runtime $O(n)$ (constant per update) and storage $O(k)$.

```

Initialize  $k$ -element array  $R$ ;
Stream  $S$ ;
for  $i = 1, \dots, k$  do
   $R[i] \leftarrow S[i]$ ;
for  $i = k + 1, \dots, \text{length}(S)$  do
   $j \leftarrow \text{random}(1, i)$ ;
  if  $j \leq k$  then
     $R[j] \leftarrow S[i]$ ;

```

Algorithm 2: Algorithm R for reservoir sampling

To ensure constant space over an unbounded stream, we draw the rejuvenation sequence $\mathcal{R}(i)$ uniformly from a reservoir. As each token of the training data is ingested by the particle filter, we decide to insert that token into the reservoir, or not, independent of the other tokens in the current document. Thus, at the end of step i of the particle filter, each of the i tokens seen so far in the training sequence has an equal probability of being in the reservoir, hence being selected for rejuvenation.

5 Experiments

We evaluate our particle filter on three datasets studied in Canini et al. (2009): `diff3`, `rel3`, and `sim3`. Each of these datasets is a collection of posts under three categories from the 20 Newsgroups dataset.¹ We use a 60% training/40% testing split of this data that is available online.²

We preprocess the data by splitting each line on non-alphabet characters, converting the resulting tokens to lower-case, and filtering out any tokens that appear in a list of common English stop words. In addition, we remove the header of every file and filter every line that does not contain a non-trailing space (which removes embedded ASCII-encoded attachments). Finally, we shuffle the order of the documents. After these steps, we compute the vocabulary for each dataset as the set of all non-singleton types in the training data augmented with a special out-of-vocabulary symbol.

¹diff3: {rec.sport.baseball, sci.space, alt.atheism}; rel3: talk.politics.{misc, guns, mideast}; and sim3: comp.{graphics, os.ms-windows.misc, windows.x}.

²<http://qwone.com/~jason/20Newsgroups/20news-bydate.tar.gz>

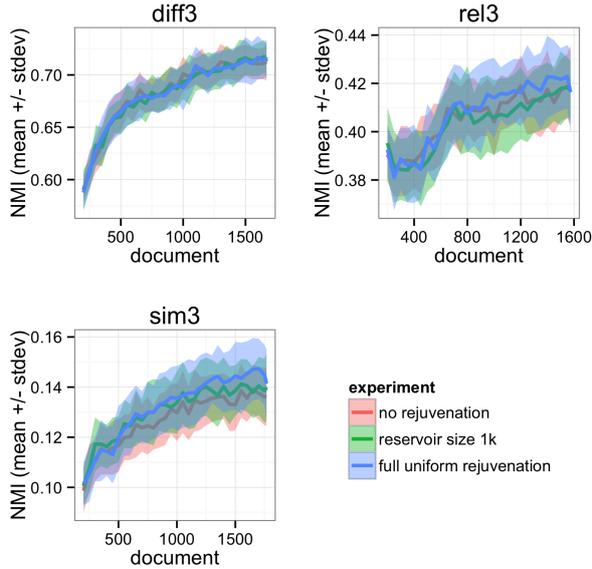


Figure 1: Fixed initialization with different reservoir sizes.

During training we report the out-of-sample NMI, calculated by holding the word proportions ϕ fixed, running five sweeps of collapsed Gibbs sampling on the test set, and computing the topic for each document as the topic assigned to the most tokens in that document. Two Gibbs sweeps have been shown to yield good performance in practice (Yao et al., 2009); we increase the number of sweeps to five after inspecting the stability on our dataset. The variance of the particle filter is often large, so for each experiment we perform 30 runs and plot the mean NMI inside bands spanning one sample standard deviation in either direction.

Fixed Initialization. Our first set of experiments has a similar parameterization³ to the experiments of Canini et al. (2009) except we draw the rejuvenation sequence from a reservoir. We initialize the particle filter with 200 Gibbs sweeps on the first 10% of each dataset. Then, for each dataset, for rejuvenation disabled, rejuvenation based on a reservoir of size 1000, and rejuvenation based on the entire history (in turn), we perform 30 runs of the particle filter from that fixed initial model. Our results (Figure 1) resemble those of Canini et al. (2009); we believe the discrepancies are mostly attributable to differences in preprocessing.

In these experiments, the initial model was not chosen arbitrarily. Rather, an initial model that yielded out-of-sample NMI close to the initial out-of-sample NMI scores reported in the previous

³ $T = 3, \alpha = \beta = 0.1, P = 100, ess = 20, |\mathcal{R}(i)| = 30$

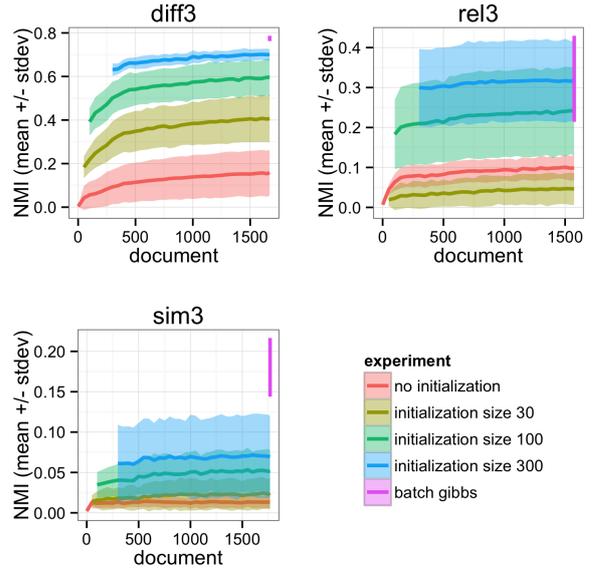


Figure 2: Variable initialization with different initialization sample sizes.

study was chosen from a set of 100 candidates.

Variable Initialization. We now investigate the significance of the initial model selection step used in the previous experiments. We run a new set of experiments in which the reservoir size is held fixed at 1000 and the size of the initialization sample is varied. Specifically, we vary the size of the initialization sample, in documents, between zero (corresponding to no Gibbs initialization), 30, 100, and 300, and also perform a run of batch Gibbs sampling (with no particle filter). In each case, 2000 Gibbs sweeps are performed. In these experiments, the initial models are not held fixed; for each of the 30 runs for each dataset, the initial model was generated by a different Gibbs chain. The results for these experiments, depicted in Figure 2, indicate that the size of the initialization sample improves mean NMI and reduces variance, and that the variance of the particle filter itself is dominated by the variance of the initial model.

Tuned Initialization. We observed previously that variance in the Gibbs initialization of the model contributes significantly to variance of the overall algorithm, as measured by NMI. With this in mind, we consider whether we can reduce variance in the initialization by tuning the initial model. Thus we perform a set of experiments in which we perform Gibbs initialization 20 times on the initialization set, setting the particle filter’s initial model to the model out of these 20 with

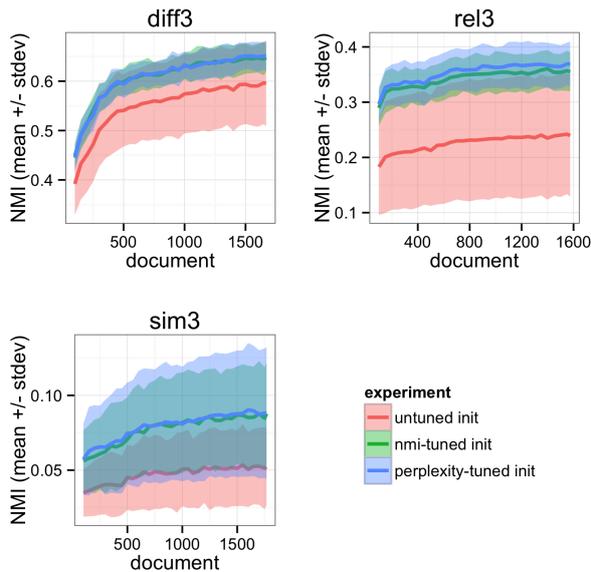


Figure 3: Variable initialization with tuning.

the highest in-sample NMI. This procedure is performed independently for each run of the particle filter. We may not always have labeled data for initialization, so we also consider a variation in which Gibbs initialization is performed 20 times on the first 80% of the initialization sample, held-out perplexity (per word) is estimated on the remaining 20%, using a first-moment particle learning approximation (Scott and Baldrige, 2013), and the particle filter is started from the model out of these 20 with the lowest held-out perplexity. The results, shown in Figure 3, show that we can ameliorate the variance due to initialization by tuning the initial model to NMI or perplexity.

6 Discussion

Motivated by a desire for cognitive plausibility, Börschinger and Johnson (2011) used a particle filter to learn Bayesian word segmentation models, following the work of Canini et al. (2009). They later showed that rejuvenation improved performance (Börschinger and Johnson, 2012), but this impaired cognitive plausibility by necessitating storage of all previous states and observations. We attempted to correct this by drawing the rejuvenation sequence from a reservoir, but our results indicate that the particle filter for LDA on our dataset is highly sensitive to initialization and not influenced by rejuvenation.

In the experiments of Börschinger and Johnson (2012), the particle cloud appears to be resampled once per utterance with a large rejuvenation se-

quence;⁴ each particle takes many more rejuvenation MCMC steps than new state transitions and thus resembles a batch MCMC sampler. In our experiments resampling is done on the order of once per document, leading to less than one rejuvenation step per transition. Future work should carefully note this ratio: sampling history much more often than new states improves performance but contradicts the intuition behind particle filters.

We have also shown that tuning the initial model using in-sample NMI or held-out perplexity can improve mean NMI and reduce variance. Perplexity (or likelihood) is often used to estimate model performance in LDA (Blei et al., 2003; Griffiths and Steyvers, 2004; Wallach et al., 2009; Hoffman et al., 2010), and does not compare the inferred model against gold-standard labels, yet it appears to be a good proxy for NMI in our experiment. Thus, if initialization continues to be crucial to performance, at least we may have the flexibility of initializing without gold-standard labels.

We have focused on NMI as our evaluation metric for comparison with Canini et al. (2009). However, evaluation of topic models is a subject of considerable debate (Wallach et al., 2009; Yao et al., 2009; Newman et al., 2010; Mimno et al., 2011) and it may be informative to investigate the effects of initialization and rejuvenation using other metrics such as perplexity or semantic coherence.

7 Conclusion

We have proposed reservoir sampling for reducing the storage complexity of a particle filter from linear to constant. This work was motivated as an expected improvement on the model of Canini et al. (2009). However, in the process of establishing an empirical baseline we discovered that rejuvenation does not play a significant role in the experiments of Canini et al. (2009). Moreover, we found that performance of the particle filter was strongly affected by the random initialization of the model, and suggested a simple approach to reduce the variability therein without using additional data. In conclusion, it is now an open question whether—and if so, under what assumptions—rejuvenation benefits particle filters for LDA and similar static Bayesian models.

Acknowledgments We thank Frank Ferraro, Keith Levin, and Mark Dredze for discussions.

⁴The ESS threshold is P ; the rejuvenation sequence is 100 or 1600 utterances, almost one sixth of the training data.

References

- Amr Ahmed, Qirong Ho, Jacob Eisenstein, Eric P. Xing, Alexander J. Smola, and Choon Hui Teo. 2011. Unified analysis of streaming news. In *Proceedings of the 20th International World Wide Web Conference (WWW)*, pages 267–276.
- Arindam Banerjee and Sugato Basu. 2007. Topic models over text streams: A study of batch and on-line unsupervised learning. In *Proceedings of the 7th SIAM International Conference on Data Mining (SDM)*, pages 431–436.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, Jan.
- Benjamin Börschinger and Mark Johnson. 2011. A particle filter algorithm for Bayesian wordsegmentation. In *Proceedings of the Australasian Language Technology Association Workshop (ALTA)*, pages 10–18.
- Benjamin Börschinger and Mark Johnson. 2012. Using rejuvenation to improve particle filtering for Bayesian word segmentation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 85–89.
- Tamara Broderick, Nicholas Boyd, Andre Wibisono, Ashia C. Wilson, and Michael I. Jordan. 2013. Streaming variational Bayes. In *Advances in Neural Information Processing Systems 26 (NIPS)*.
- Kevin R. Canini, Lei Shi, and Thomas L. Griffiths. 2009. Online inference of topics with latent Dirichlet allocation. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*.
- Arnaud Doucet, Nando de Freitas, Kevin Murphy, and Stuart Russell. 2000. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 176–183.
- Arnaud Doucet, Nando de Freitas, and Neil Gordon, editors. 2001. *Sequential Monte Carlo Methods in Practice*. Springer, New York.
- Walter R. Gilks and Carlo Berzuini. 2001. Following a moving target—Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society*, 63(1):127–146.
- Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5228–5235, Apr.
- Matthew D. Hoffman, David M. Blei, and Francis Bach. 2010. Online learning for latent Dirichlet allocation. In *Advances in Neural Information Processing Systems 23 (NIPS)*.
- Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. 2013. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347, May.
- Jun S. Liu and Rong Chen. 1998. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443):1032–1044, Sep.
- David Mimno, Hanna M. Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 2011. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*, pages 262–272.
- David Mimno, Matthew D. Hoffman, and David M. Blei. 2012. Sparse stochastic inference for latent Dirichlet allocation. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*.
- Thomas Minka and John Lafferty. 2002. Expectation-propagation for the generative aspect model. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 352–359.
- David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. 2010. Automatic evaluation of topic coherence. In *Human Language Technologies: 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT)*, pages 100–108.
- Michael K. Pitt and Neil Shephard. 1999. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, Jun.
- James G. Scott and Jason Baldridge. 2013. A recursive estimate for the predictive likelihood in a topic model. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics (AISTATS)*.
- Jeffrey S. Vitter. 1985. Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, 11(1):37–57, Mar.
- Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. 2009. Evaluation methods for topic models. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*.
- Limin Yao, David Mimno, and Andrew McCallum. 2009. Efficient methods for topic model inference on streaming document collections. In *15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 937–946.

Comparing Automatic Evaluation Measures for Image Description

Desmond Elliott and Frank Keller

Institute for Language, Cognition, and Computation
School of Informatics, University of Edinburgh
d.elliott@ed.ac.uk, keller@inf.ed.ac.uk

Abstract

Image description is a new natural language generation task, where the aim is to generate a human-like description of an image. The evaluation of computer-generated text is a notoriously difficult problem, however, the quality of image descriptions has typically been measured using unigram BLEU and human judgements. The focus of this paper is to determine the correlation of automatic measures with human judgements for this task. We estimate the correlation of unigram and Smoothed BLEU, TER, ROUGE-SU4, and Meteor against human judgements on two data sets. The main finding is that unigram BLEU has a weak correlation, and Meteor has the strongest correlation with human judgements.

1 Introduction

Recent advances in computer vision and natural language processing have led to an upsurge of research on tasks involving both vision and language. State of the art visual detectors have made it possible to hypothesise *what* is in an image (Guillaumin et al., 2009; Felzenszwalb et al., 2010), paving the way for automatic image description systems. The aim of such systems is to extract and reason about visual aspects of images to generate a human-like description. An example of the type of image and gold-standard descriptions available can be seen in Figure 1. Recent approaches to this task have been based on slot-filling (Yang et al., 2011; Elliott and Keller, 2013), combining web-scale n-grams (Li et al., 2011), syntactic tree substitution (Mitchell et al., 2012), and description-by-retrieval (Farhadi et al., 2010; Ordonez et al., 2011; Hodosh et al., 2013). Image description has been compared to translating an image into text (Li et al., 2011; Kulkarni et al., 2011) or summarising an image



1. An older woman with a small dog in the snow.
2. A woman and a cat are outside in the snow.
3. A woman in a brown vest is walking on the snow with an animal.
4. A woman with a red scarf covering her head walks with her cat on snow-covered ground.
5. Heavy set woman in snow with a cat.

Figure 1: An image from the Flickr8K data set and five human-written descriptions. These descriptions vary in the adjectives or prepositional phrases that describe the woman (1, 3, 4, 5), incorrect or uncertain identification of the cat (1, 3), and include a sentence without a verb (5).

(Yang et al., 2011), resulting in the adoption of the evaluation measures from those communities.

In this paper we estimate the correlation of human judgements with five automatic evaluation measures on two image description data sets. Our work extends previous studies of evaluation measures for image description (Hodosh et al., 2013), which focused on unigram-based measures and reported agreement scores such as Cohen's κ rather than correlations. The main finding of our analysis is that TER and unigram BLEU are weakly corre-

lated against human judgements, ROUGE-SU4 and Smoothed BLEU are moderately correlated, and the strongest correlation is found with Meteor.

2 Methodology

We estimate Spearman’s ρ for five different automatic evaluation measures against human judgements for the automatic image description task. Spearman’s ρ is a non-parametric correlation coefficient that restricts the ability of outlier data points to skew the co-efficient value. The automatic measures are calculated on the sentence level and correlated against human judgements of semantic correctness.

2.1 Data

We perform the correlation analysis on the Flickr8K data set of Hodosh et al. (2013), and the data set of Elliott and Keller (2013).

The test data of the Flickr8K data set contains 1,000 images paired with five reference descriptions. The images were retrieved from Flickr, the reference descriptions were collected from Mechanical Turk, and the human judgements were collected from expert annotators as follows: each image in the test data was paired with the highest scoring sentence(s) retrieved from all possible test sentences by the TRI5SEM model in Hodosh et al. (2013). Each image–description pairing in the test data was judged for semantic correctness by three expert human judges on a scale of 1–4. We calculate automatic measures for each image–retrieved sentence pair against the five reference descriptions for the original image.

The test data of Elliott and Keller (2013) contains 101 images paired with three reference descriptions. The images were taken from the PAS-CAL VOC Action Recognition Task, the reference descriptions were collected from Mechanical Turk, and the judgements were also collected from Mechanical Turk. Elliott and Keller (2013) generated two-sentence descriptions for each of the test images using four variants of a slot-filling model, and collected five human judgements of the semantic correctness and grammatical correctness of the description on a scale of 1–5 for each image–description pair, resulting in a total of 2,042 human judgement–description pairings. In this analysis, we use only the first sentence of the description, which describes the event depicted in the image.

2.2 Automatic Evaluation Measures

BLEU measures the effective overlap between a reference sentence X and a candidate sentence Y . It is defined as the geometric mean of the effective n-gram precision scores, multiplied by the brevity penalty factor BP to penalise short translations. p_n measures the effective overlap by calculating the proportion of the maximum number of n-grams co-occurring between a candidate and a reference and the total number of n-grams in the candidate text. More formally,

$$BLEU = BP \cdot \exp \left(\sum_{n=1}^N w_n \log p_n \right)$$

$$p_n = \frac{\sum_{c \in cand} \sum_{ngram \in c} count_{clip}(ngram)}{\sum_{c \in cand} \sum_{ngram \in c} count(ngram)}$$

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

Unigram BLEU without a brevity penalty has been reported by Kulkarni et al. (2011), Li et al. (2011), Ordonez et al. (2011), and Kuznetsova et al. (2012); to the best of our knowledge, the only image description work to use higher-order n-grams with BLEU is Elliott and Keller (2013). In this paper we use the smoothed BLEU implementation of Clark et al. (2011) to perform a sentence-level analysis, setting $n = 1$ and no brevity penalty to get the unigram BLEU measure, or $n = 4$ with the brevity penalty to get the Smoothed BLEU measure. We note that a higher BLEU score is better.

ROUGE measures the longest common subsequence of tokens between a candidate Y and reference X . There is also a variant that measures the co-occurrence of pairs of tokens in both the candidate and reference (a skip-bigram): ROUGE-SU*. The skip-bigram calculation is parameterised with d_{skip} , the maximum number of tokens between the words in the skip-bigram. Setting d_{skip} to 0 is equivalent to bigram overlap and setting d_{skip} to ∞ means tokens can be any distance apart. If $\alpha = |SKIP2(X, Y)|$ is the number of matching skip-bigrams between the reference and the candidate, then skip-bigram ROUGE is formally defined as:

$$R_{SKIP2} = \alpha / \binom{\alpha}{2}$$

ROUGE has been used by only Yang et al. (2011) to measure the quality of generated descriptions, using a variant they describe as ROUGE-1. We set $d_{skip} = 4$ and award partial credit for unigram only matches, otherwise known as ROUGE-SU4. We use ROUGE v.1.5.5 for the analysis, and configure the evaluation script to return the result for the average score for matching between the candidate and the references. A higher ROUGE score is better.

TER measures the number of modifications a human would need to make to transform a candidate Y into a reference X . The modifications available are insertion, deletion, substitute a single word, and shift a word an arbitrary distance. TER is expressed as the percentage of the sentence that needs to be changed, and can be greater than 100 if the candidate is longer than the reference. More formally,

$$TER = \frac{|\text{edits}|}{|\text{reference tokens}|}$$

TER has not yet been used to evaluate image description models. We use v.0.8.0 of the TER evaluation tool, and a lower TER is better.

Meteor is the harmonic mean of unigram precision and recall that allows for exact, synonym, and paraphrase matchings between candidates and references. It is calculated by generating an alignment between the tokens in the candidate and reference sentences, with the aim of a 1:1 alignment between tokens and minimising the number of chunks ch of contiguous and identically ordered tokens in the sentence pair. The alignment is based on exact token matching, followed by Wordnet synonyms, and then stemmed tokens. We can calculate precision, recall, and F-measure, where m is the number of aligned unigrams between candidate and reference. Meteor is defined as:

$$M = (1 - Pen) \cdot F_{mean}$$

$$Pen = \gamma \left(\frac{ch}{m} \right)^\theta$$

$$F_{mean} = \frac{PR}{\alpha P + (1 - \alpha)R}$$

$$P = \frac{|m|}{|\text{unigrams in candidate}|}$$

$$R = \frac{|m|}{|\text{unigrams in reference}|}$$

We calculated the Meteor scores using release 1.4.0 with the package-provided free parameter settings of 0.85, 0.2, 0.6, and 0.75 for the matching components. Meteor has not yet been reported to evaluate

	Flickr 8K co-efficient $n = 17,466$	E&K (2013) co-efficient $n = 2,040$
METEOR	0.524	0.233
ROUGE SU-4	0.435	0.188
Smoothed BLEU	0.429	0.177
Unigram BLEU	0.345	0.097
TER	-0.279	-0.044

Table 1: Spearman’s correlation co-efficient of automatic evaluation measures against human judgements. All correlations are significant at $p < 0.001$.

the performance of different models on the image description task; a higher Meteor score is better.

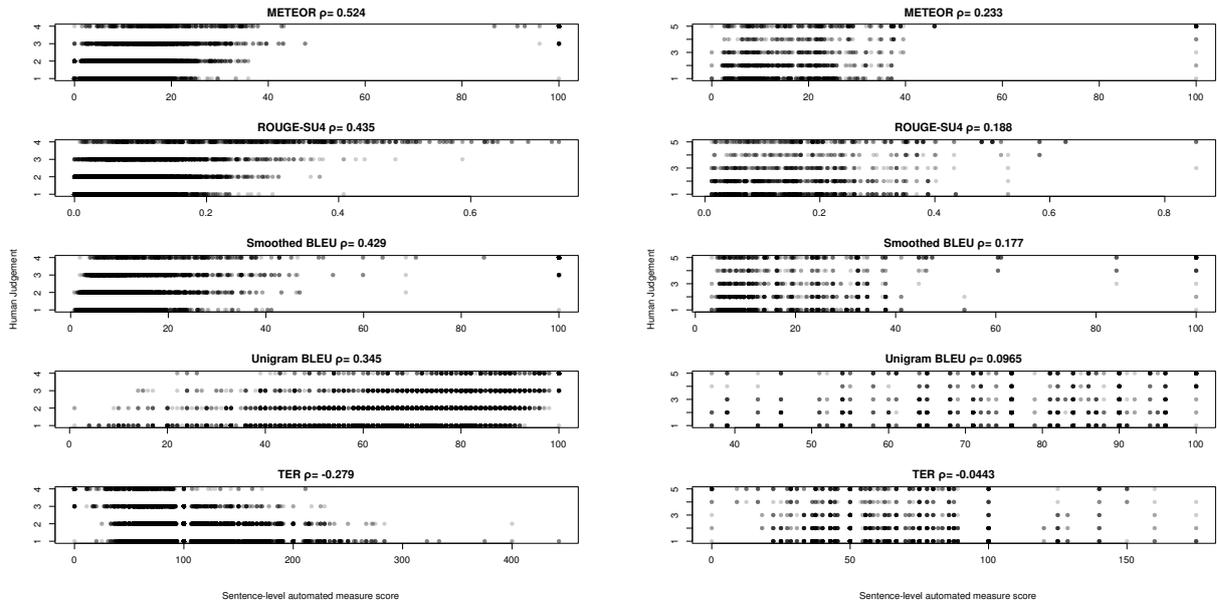
2.3 Protocol

We performed the correlation analysis as follows. The sentence-level evaluation measures were calculated for each image–description–reference tuple. We collected the BLEU, TER, and Meteor scores using MultEval (Clark et al., 2011), and the ROUGE-SU4 scores using the RELEASE-1.5.5.pl script. The evaluation measure scores were then compared with the human judgements using Spearman’s correlation estimated at the sentence-level.

3 Results

Table 1 shows the correlation co-efficients between automatic measures and human judgements and Figures 2(a) and (b) show the distribution of scores for each measure against human judgements. To classify the strength of the correlations, we followed the guidance of Dancey and Reidy (2011), who posit that a co-efficient of 0.0–0.1 is uncorrelated, 0.11–0.4 is *weak*, 0.41–0.7 is *moderate*, 0.71–0.90 is *strong*, and 0.91–1.0 is *perfect*.

On the Flickr8k data set, all evaluation measures can be classified as either *weakly* correlated or *moderately* correlated with human judgements and all results are significant. TER is only weakly correlated with human judgements but could prove useful in comparing the types of differences between models. An analysis of the distribution of TER scores in Figure 2(a) shows that differences in candidate and reference length are prevalent in the image description task. Unigram BLEU is also only weakly correlated against human judgements, even though it has been reported extensively for this task.



(a) Flick8K data set, n=17,466.

(b) E&K (2013) data set, n=2,042.

Figure 2: Distribution of automatic evaluation measures against human judgements. ρ is the correlation between human judgements and the automatic measure. The intensity of each point indicates the number of occurrences that fall into that range.

Figure 2(a) shows an almost uniform distribution of unigram BLEU scores, regardless of the human judgement. Smoothed BLEU and ROUGE-SU4 are moderately correlated with human judgements, and the correlation is stronger than with unigram BLEU. Finally, Meteor is most strongly correlated measure against human judgements. A similar pattern is observed in the Elliott and Keller (2013) data set, though the correlations are lower across all measures. This could be caused by the smaller sample size or because the descriptions were generated by a computer, and not retrieved from a collection of human-written descriptions containing the gold-standard text, as in the Flickr8K data set.

Qualitative Analysis

Figure 3 shows two images from the test collection of the Flickr8K data set with a low Meteor score and a maximum human judgement of semantic correctness. The main difference between the candidates and references are in deciding *what* to describe (content selection), and *how* to describe it (realisation). We can hypothesise that in both translation and summarisation, the source text acts as a lexical and semantic framework within which the translation or summarisation process takes place. In Figure 3(a), the authors of the descriptions made different decisions on *what* to describe. A decision

has been made to describe the role of the officials in the candidate text, and not in the reference text. The underlying cause of this is an active area of research in the human vision literature and can be attributed to bottom-up effects, such as saliency (Itti et al., 1998), top-down contextual effects (Torralba et al., 2006), or rapidly-obtained scene properties (Oliva and Torralba, 2001). In (b), we can see the problem of deciding how to describe the selected content. The reference uses a more specific noun to describe the person on the bicycle than the candidate.

4 Discussion

There are several differences between our analysis and that of Hodosh et al. (2013). First, we report Spearman’s ρ correlation coefficient of automatic measures against human judgements, whereas they report agreement between judgements and automatic measures in terms of Cohen’s κ . The use of κ requires the transformation of real-valued scores into categorical values, and thus loses information; we use the judgement and evaluation measure scores in their original forms. Second, our use of Spearman’s ρ means we can readily use all of the available data for the correlation analysis, whereas Hodosh et al. (2013) report agreement on thresholded subsets of the data. Third, we report the correlation coefficients against five evaluation measures,



Candidate: Football players gathering to contest something to collaborating officials.

Reference: A football player in red and white is holding both hands up.

(a)



Candidate: A man is attempting a stunt with a bicycle.

Reference: Bmx biker Jumps off of ramp.

(b)

Figure 3: Examples in the test data with low Meteor scores and the maximum expert human judgement. (a) the candidate and reference are from the same image, and show differences in *what* to describe, in (b) the descriptions are retrieved from different images and show differences in *how* to describe an image.

some of which go beyond unigram matchings between references and candidates, whereas they only report unigram BLEU and unigram ROUGE. It is therefore difficult to directly compare the results of our correlation analysis against Hodosh et al.’s agreement analysis, but they also reach the conclusion that unigram BLEU is not an appropriate measure of image description performance. However, we do find stronger correlations with Smoothed BLEU, skip-bigram ROUGE, and Meteor.

In contrast to the results presented here, Reiter and Belz (2009) found no significant correlations of automatic evaluation measures against human judgements of the *accuracy* of machine-generated weather forecasts. They did, however, find significant correlations of automatic measures against *fluency* judgements. There are no fluency judgements available for Flickr8K, but Elliott and Keller (2013) report grammaticality judgements for their data, which are comparable to fluency ratings. We failed to find significant correlations between grammaticality judgements and any of the automatic measures on the Elliott and Keller (2013) data. This discrepancy could be explained in terms of the differences between the weather forecast generation and image description tasks, or because the image description data sets contain thousands of texts and a few human judgements per text, whereas the data sets of Reiter and Belz (2009) included hundreds of texts with 30 human judges.

5 Conclusions

In this paper we performed a sentence-level correlation analysis of automatic evaluation measures against expert human judgements for the automatic image description task. We found that sentence-level unigram BLEU is only weakly correlated with human judgements, even though it has extensively reported in the literature for this task. Meteor was found to have the highest correlation with human judgements, but it requires Wordnet and paraphrase resources that are not available for all languages. Our findings held when judgements were made on human-written or computer-generated descriptions.

The variability in what and how people describe images will cause problems for all of the measures compared in this paper. Nevertheless, we propose that unigram BLEU should no longer be used as an objective function for automatic image description because it has a weak correlation with human accuracy judgements. We recommend adopting either Meteor, Smoothed BLEU, or ROUGE-SU4 because they show stronger correlations with human judgements. We believe these suggestions are also applicable to the ranking tasks proposed in Hodosh et al. (2013), where automatic evaluation scores could act as features to a ranking function.

Acknowledgments

Alexandra Birch and R. Calen Walshe, and the anonymous reviewers provided valuable feedback on this paper. The research is funded by ERC Starting Grant SYNPROC No. 203427.

References

- Jonathon H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 176–181, Portland, Oregon, USA.
- Christine Dancey and John Reidy, 2011. *Statistics Without Maths for Psychology*, page 175. Prentice Hall, 5th edition.
- Desmond Elliott and Frank Keller. 2013. Image Description using Visual Dependency Representations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302, Seattle, Washington, U.S.A.
- Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. 2010. Every picture tells a story: generating sentences from images. In *Proceedings of the 11th European Conference on Computer Vision*, pages 15–29, Heraklion, Crete, Greece.
- Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. 2010. Object Detection with Discriminatively Trained Part-Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645.
- Matthieu Guillaumin, Thomas Mensink, Jakob J. Verbeek, and Cornelia Schmid. 2009. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *IEEE 12th International Conference on Computer Vision*, pages 309–316, Kyoto, Japan.
- Micah Hodosh, Peter Young, and Julia Hockenmaier. 2013. Framing Image Description as a Ranking Task : Data , Models and Evaluation Metrics. *Journal of Artificial Intelligence Research*, 47:853–899.
- Laurent Itti, Christof Koch, and Ernst Niebur. 1998. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259.
- Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C. Berg, and Tamara L. Berg. 2011. Baby talk: Understanding and generating simple image descriptions. In *The 24th IEEE Conference on Computer Vision and Pattern Recognition*, pages 1601–1608, Colorado Springs, Colorado, U.S.A.
- Polina Kuznetsova, Vicente Ordonez, Alexander C. Berg, Tamara L. Berg, and Yejin Choi. 2012. Collective Generation of Natural Image Descriptions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 359–368, Jeju Island, South Korea.
- Siming Li, Girish Kulkarni, Tamara L. Berg, Alexander C. Berg, and Yejin Choi. 2011. Composing simple image descriptions using web-scale n-grams. In *Fifteenth Conference on Computational Natural Language Learning*, pages 220–228, Portland, Oregon, U.S.A.
- Margaret Mitchell, Jesse Dodge, Amit Goyal, Kota Yamaguchi, Karl Stratos, Alyssa Mensch, Alex Berg, Tamara Berg, and Hal Daumé III. 2012. Midge : Generating Image Descriptions From Computer Vision Detections. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 747–756, Avignon, France.
- Aude Oliva and Antonio Torralba. 2001. Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope. *International Journal of Computer Vision*, 42(3):145–175.
- Vicente Ordonez, Girish Kulkarni, and Tamara L. Berg. 2011. Im2Text: Describing Images Using 1 Million Captioned Photographs. In *Advances in Neural Information Processing Systems 24*, Granada, Spain.
- Ehud Reiter and A Belz. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics*, 35(4):529–558.
- Antonio Torralba, Aude Oliva, Monica S. Castelano, and John M. Henderson. 2006. Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. *Psychological Review*, 113(4):766–786.
- Yezhou Yang, Ching Lik Teo, Hal Daumé III, and Yiannis Aloimonos. 2011. Corpus-Guided Sentence Generation of Natural Images. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 444–454, Edinburgh, Scotland, UK.

Learning a Lexical Simplifier Using Wikipedia

Colby Horn, Cathryn Manduca and David Kauchak

Computer Science Department
Middlebury College

{chorn, cmanduca, dkauchak}@middlebury.edu

Abstract

In this paper we introduce a new lexical simplification approach. We extract over 30K candidate lexical simplifications by identifying aligned words in a sentence-aligned corpus of English Wikipedia with Simple English Wikipedia. To apply these rules, we learn a feature-based ranker using SVM^{rank} trained on a set of labeled simplifications collected using Amazon’s Mechanical Turk. Using human simplifications for evaluation, we achieve a precision of 76% with changes in 86% of the examples.

1 Introduction

Text simplification is aimed at reducing the reading and grammatical complexity of text while retaining the meaning (Chandrasekar and Srinivas, 1997). Text simplification techniques have a broad range of applications centered around increasing data availability to both targeted audiences, such as children, language learners, and people with cognitive disabilities, as well as to general readers in technical domains such as health and medicine (Feng, 2008).

Simplifying a text can require a wide range of transformation operations including lexical changes, syntactic changes, sentence splitting, deletion and elaboration (Coster and Kauchak, 2011; Zhu et al., 2010). In this paper, we examine a restricted version of the text simplification problem, lexical simplification, where text is simplified by substituting words or phrases with simpler variants. Even with this restriction, lexical simplification techniques have been shown to positively impact the simplicity of text and to improve reader understanding and information retention (Leroy et al., 2013). Additionally, restricting the set of transformation operations allows for

more straightforward evaluation than the general simplification problem (Specia et al., 2012).

Most lexical simplification techniques rely on transformation rules that change a word or phrase into a simpler variant with similar meaning (Biran et al., 2011; Specia et al., 2012; Yatskar et al., 2010). Two main challenges exist for this type of approach. First, the lexical focus of the transformation rules makes generalization difficult; a large number of transformation rules is required to achieve reasonable coverage and impact. Second, rules do not apply in all contexts and care must be taken when performing lexical transformations to ensure local cohesion, grammaticality and, most importantly, the preservation of the original meaning.

In this paper, we address both of these issues. We leverage a data set of 137K aligned sentence pairs between English Wikipedia and Simple English Wikipedia to learn simplification rules. Previous approaches have used unaligned versions of Simple English Wikipedia to learn rules (Biran et al., 2011; Yatskar et al., 2010), however, by using the aligned version we are able to learn a much larger rule set.

To apply lexical simplification rules to a new sentence, a decision must be made about which, if any, transformations should be applied. Previous approaches have used similarity measures (Biran et al., 2011) and feature-based approaches (Specia et al., 2012) to make this decision. We take the latter approach and train a supervised model to rank candidate transformations.

2 Problem Setup

We learn lexical simplification rules that consist of a word to be simplified and a list of candidate simplifications:

$$w \rightarrow c_1, c_2, \dots, c_m$$

Consider the two aligned sentence pairs in Table

The first school was established in 1857.
The first school was started in 1857.
The district was established in 1993 by merging the former districts of Bernau and Eberswalde.
The district was made in 1993 by joining the old districts of Bernau and Eberswalde.

Table 1: Two aligned sentence pairs. The bottom sentence is a human simplified version of the top sentence. Bold words are candidate lexical simplifications.

1. The bottom sentence of each pair is a simplified variant of the top sentence. By identifying aligned words within the aligned sentences, candidate lexical simplifications can be learned. The bold words show two such examples, though other candidates exist in the bottom pair. By examining aligned sentence pairs we can learn a simplification rule. For example, we might learn:

established \rightarrow *began, made, settled, started*

Given a sentence s_1, s_2, \dots, s_n , a simplification rule applies if the left hand side of the rule can be found in the sentence ($s_i = w$, for some i). If a rule applies, then a decision must be made about which, if any, of the candidate simplifications should be substituted for the word w to simplify the sentence. For example, if we were attempting to simplify the sentence

The ACL was established in 1962.

using the simplification rule above, some of the simplification options would not apply because of grammatical constraints, e.g. *began*, while others would not apply for semantic reasons, e.g. *settled*. This does not mean that these are not good simplifications for *established* since in other contexts, they might be appropriate. For example, in the sentence

The researcher established a new paper writing routine.

began is a reasonable option.

3 Learning a Lexical Simplifier

We break the learning problem into two steps: 1) learn a set of simplification rules and 2) learn a ranking function for determining the best simplification candidate when a rule applies. Each of these steps are outlined below.

3.1 Rule Extraction

To extract the set of simplification rules, we use a sentence-aligned data set of English Wikipedia sentences (referred to as *normal*) aligned to Simple English Wikipedia sentences (referred to as *simple*) (Coster and Kauchak, 2011). The data set contains 137K such aligned sentence pairs.

Given a normal sentence and the corresponding aligned simple sentence, candidate simplifications are extracted by identifying a word in the simple sentence that corresponds to a different word in the normal sentence. To identify such pairs, we automatically induce a word alignment between the normal and simple sentence pairs using GIZA++ (Och and Ney, 2000). Words that are aligned are considered as possible candidates for extraction. Due to errors in the sentence and word alignment processes, not all words that are aligned are actually equivalent lexical variants. We apply the following filters to reduce such spurious alignments:

- We remove any pairs where the normal word occurs in a stoplist. Stoplist words tend to be simple already and stoplist words that are being changed are likely either bad alignments or are not simplifications.
- We require that the part of speeches (POS) of the two words be the same. The parts of speech were calculated based on a full parse of the sentences using the Berkeley parser (Petrov and Klein, 2007).
- We remove any candidates where the POS is labeled as a proper noun. In most cases, proper nouns should not be simplified.

All other aligned word pairs are extracted. To generate the simplification rules, we collect all candidate simplifications (simple words) that are aligned to the same normal word.

As mentioned before, one of the biggest challenges for lexical simplification systems is generalizability. To improve the generalizability of the extracted rules, we add morphological variants of the words in the rules. For nouns, we include both singular and plural variants. For verbs, we expand to all inflection variants. The morphological changes are generated using MorphAdorner (Burns, 2013) and are applied symmetrically: any change to the normal word is also applied to the corresponding simplification candidates.

3.2 Lexical Simplification as a Ranking Problem

A lexical simplification example consists of three parts: 1) a sentence, s_1, s_2, \dots, s_n , 2) a word in that sentence, s_i , and 3) a list of candidate simplifications for s_i , c_1, c_2, \dots, c_m . A labeled example is an example where the rank of the candidate simplifications has been specified. Given a set of labeled examples, the goal is to learn a ranking function that, given an unlabeled example (example without the candidate simplifications ranked), specifies a ranking of the candidates.

To learn this function, features are extracted from a set of labeled lexical simplification examples. These labeled examples are then used to train a ranking function. We use SVM^{rank} (Joachims, 2006), which uses a linear support vector machine.

Besides deciding which of the candidates is most applicable in the context of the sentence, even if a rule applies, we must also decide if any simplification should occur. For example, there may be an instance where none of the candidate simplifications are appropriate in this context. Rather than viewing this as a separate problem, we incorporate this decision into the ranking problem by adding w as a candidate simplification. For each rule, $w \rightarrow c_1, c_2, \dots, c_m$ we add one additional candidate simplification which does not change the sentence, $w \rightarrow c_1, c_2, \dots, c_m, w$. If w is ranked as the most likely candidate by the ranking algorithm, then the word is not simplified.

3.2.1 Features

The role of the features is to capture information about the applicability of the word in the context of the sentence as well as the simplicity of the word. Many features have been suggested previously for use in determining the simplicity of a word (Specia et al., 2012) and for determining if a word is contextually relevant (Biran et al., 2011; McCarthy and Navigli, 2007). Our goal for this paper is not feature exploration, but to examine the usefulness of a general framework for feature-based ranking for lexical simplification. The features below represent a first pass at candidate features, but many others could be explored.

Candidate Probability

$p(c_i|w)$: in the sentence-aligned Wikipedia data, when w is aligned to some candidate simplification, what proportion of the time is that candidate c_i .

Frequency

The frequency of a word has been shown to correlate with the word’s simplicity and with people’s knowledge of that word (Leroy and Kauchak, 2013). We measured a candidate simplification’s frequency in two corpora: 1) Simple English Wikipedia and 2) the web, as measured by the unigram frequency from the Google n-gram corpus (Brants and Franz, 2006).

Language Models

n -gram language models capture how likely a particular sequence is and can help identify candidate simplifications that are not appropriate in the context of the sentence. We included features from four different language models trained on four different corpora: 1) Simple English Wikipedia, 2) English Wikipedia, 3) Google n-gram corpus and 4) a linearly interpolated model between 1) and 2) with $\lambda = 0.5$, i.e. an even blending. We used the SRI language modeling toolkit (Stolcke, 2002) with Kneser-Kney smoothing. All models were trigram language models except the Google n-gram model, which was a 5-gram model.

Context Frequency

As another measure of the applicability of a candidate in the context of the sentence, we also calculate the frequency in the Google n-grams of the candidate simplification in the context of the sentence with context windows of one and two words. If the word to be substituted is at position i in the sentence ($w = s_i$), then the one word window frequency for simplification c_j is the trigram frequency of $s_{i-1} c_j s_{i+1}$ and the two word window the 5-gram frequency of $s_{i-2} s_{i-1} c_j s_{i+1} s_{i+2}$.

4 Data

For training and evaluation of the models, we collected human labelings of 500 lexical simplification examples using Amazon’s Mechanical Turk (MTurk)¹. MTurk has been used extensively for annotating and evaluating NLP tasks and has been shown to provide data that is as reliable as other forms of human annotation (Callison-Burch and Dredze, 2010; Zaidan and Callison-Burch, 2011).

Figure 1 shows an example of the task we asked annotators to do. Given a sentence and a word to be simplified, the task is to suggest a simpler variant of that word that is appropriate in the context of the sentence. Candidate sentences were se-

¹<https://www.mturk.com/>

Enter a *simpler* word that could be substituted for the red, bold word in the sentence. A *simpler* word is one that would be understood by more people or people with a lower reading level (e.g. children).

Food is procured with its suckers and then crushed using its tough “beak” of chitin.

Figure 1: Example task setup on MTurk soliciting lexical simplifications from annotators.

lected from the sentence-aligned Wikipedia corpus where a word in the normal sentence is being simplified to a different word in the simple sentence, as identified by the automatically induced word alignment. The normal sentence and the aligned word were then selected for annotation. These examples represent words that other people (those that wrote/edited the Simple English Wikipedia page) decided were difficult and required simplification.

We randomly selected 500 such sentences and collected candidate simplifications from *50 people per sentence*, for a total of 25,000 annotations. To participate in the annotation process, we required that the MTurk workers live in the U.S. (for English proficiency) and had at least a 95% acceptance rate on previous tasks.

The simplifications suggested by the annotators were then tallied and the resulting list of simplifications with frequencies provides a ranking for training the candidate ranker. Table 2 shows the ranked list of annotations collected for the example in Figure 1. This data set is available online.²

Since these examples were selected from English Wikipedia they, and the corresponding aligned Simple English Wikipedia sentences, were removed from *all* resources used during both the rule extraction and the training of the ranker.

5 Experiments

5.1 Other Approaches

We compared our lexical simplification approach (**rank-simplify**) to two other approaches. To understand the benefit of the feature-based ranking algorithm, we compared against a simplifier that uses the same rule set, but ranks the candidates only based on their frequency in Simple English Wikipedia (**frequency**). This is similar to baselines used in previous work (Biran et al., 2011).

To understand how our extracted rules compared to the rules extracted by Biran et al., we

²<http://www.cs.middlebury.edu/~dkauchak/simplification/>

used their rules with *our* ranking approach (**rank-Biran**). Their approach also extracts rules from a corpus of English Wikipedia and Simple English Wikipedia, however, they do not utilize a sentence-aligned version and instead rely on context similarity measures to extract their rules.

5.2 Evaluation

We used the 500 ranked simplification examples to train and evaluate our approach. We employed 10-fold cross validation for all experiments, training on 450 examples and testing on 50.

We evaluated the models with four different metrics:

precision: Of the words that the system changed, what percentage were found in *any* of the human annotations.

precision@*k*: Of the words that the system changed, what percentage were found in the top *k* human annotations, where the annotations were ranked by response frequency. For example, if we were calculating the precision@1 for the example in Table 2, only “obtained” would be considered correct.

accuracy: The percentage of the test examples where the system made a change to one of the annotations suggested by the human annotators. Note that unlike precision, if the system does not suggest a change to a word that was simplified it still gets penalized.

changed: The percentage of the test examples where the system suggested some change (even if it wasn’t a “correct” change).

5.3 Results

Table 3 shows the precision, accuracy and percent changed for the three systems. Based on all three metrics, our system achieves the best results. Although the rules generated by Biran et al. have reasonable precision, they suffer from a lack of coverage, only making changes on about 5% of the

word	frequency	word	frequency	word	frequency
obtained	17	made	2	secured	1
gathered	9	created	1	found	1
gotten	8	processed	1	attained	1
grabbed	4	received	1	procured	1
acquired	2	collected	1	aquired	1

Table 2: Candidate simplifications generated using MTurk for the examples in Figure 1. The frequency is the number of annotators that suggested that simplification.

	precision	accuracy	changed
frequency	53.9%	46.1%	84.9%
rank-Biran	71.4%	3.4%	5.2%
rank-simplify	76.1%	66.3%	86.3%

Table 3: Precision, accuracy and percent changed for the three systems, averaged over the 10 folds.

examples. For our approach, the extracted rules had very good coverage, applying in over 85% of the examples.

This difference in coverage can be partially attributed to the number of rules learned. We learned simplifications for 14,478 words with an average of 2.25 candidate simplifications per word. In contrast, the rules from Biran et al. only had simplifications for 3,598 words with an average of 1.18 simplifications per word.

The precision of both of the approaches that utilized the SVM candidate ranking were significantly better than the frequency-based approach. To better understand the types of suggestions made by the systems, Figure 2 shows the precision@ k for increasing k . On average, over the 500 examples we collected, people suggested 12 different simplifications, though this varied depending on the word in question and the sentence. As such, at around $k=12$, the precision@ k of most of the systems has almost reached the final precision. However, even at $k = 5$, which only counts correct an answer in the top 5 human suggested results, our system still achieved a precision of around 67%.

6 Future Work

In this paper we have introduced a new rule extraction algorithm and a new feature-based ranking approach for applying these rules in the context of different sentences. The number of rules learned is an order of magnitude larger than any previous lexical simplification approach and the

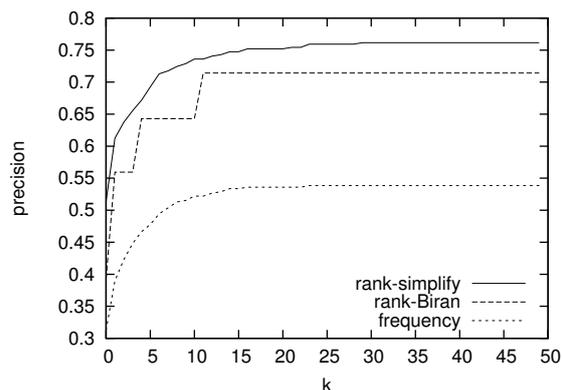


Figure 2: Precision@ k for varying k for the three different approaches averaged over the 10 folds.

quality of the resulting simplifications after applying these rules is better than previous approaches.

Many avenues exist for improvement and for better understanding how well the current approach works. First, we have only explored a small set of possible features in the ranking algorithm. Additional improvements could be seen by incorporating a broader feature set. Second, more analysis needs to be done to understand the quality of the produced simplifications and their impact on the simplicity of the resulting sentences. Third, the experiments above assume that the word to be simplified has already been identified in the sentence. This identification step also needs to be explored to implement a sentence-level simplifier using our approach. Fourth, the ranking algorithm can be applied to most simplification rules (e.g. we applied the ranking approach to the rules obtained by Biran et al. (2011)). We hope to explore other approaches for increasing the rule set by incorporating other rule sources and other rule extraction techniques.

References

- Or Biran, Samuel Brody, and Noemie Elhadad. 2011. Putting it simply: A context-aware approach to lexical simplification. In *Proceedings of ACL*.
- Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram version 1. Linguistic Data Consortium, Philadelphia.
- Philip R. Burns. 2013. Morphadorner v2: A Java library for the morphological adornment of english language texts.
- Chris Callison-Burch and Mark Dredze. 2010. Creating speech and language data with Amazon's Mechanical Turk. In *Proceedings of NAACL-HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*.
- Raman Chandrasekar and Bangalore Srinivas. 1997. Automatic induction of rules for text simplification. In *Knowledge Based Systems*.
- William Coster and David Kauchak. 2011. Simple English Wikipedia: A new text simplification task. In *Proceedings of ACL*.
- Lijun Feng. 2008. Text simplification: A survey. CUNY Technical Report.
- Thorsten Joachims. 2006. Training linear svms in linear time. In *Proceedings of KDD*.
- Gondy Leroy and David Kauchak. 2013. The effect of word familiarity on actual and perceived text difficulty. *Journal of American Medical Informatics Association*.
- Gondy Leroy, James E. Endicott, David Kauchak, Obay Mouradi, and Melissa Just. 2013. User evaluation of the effects of a text simplification algorithm using term familiarity on perception, understanding, learning, and information retention. *Journal of Medical Internet Research (JMIR)*.
- Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of SEMEVAL*.
- F. J. Och and H. Ney. 2000. Improved statistical alignment models. In *ACL*.
- Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of HTL-NAACL*.
- Lucia Specia, Sujay Kumar Jauhar, and Rada Mihalcea. 2012. Semeval-2012 task 1: English lexical simplification. In *Joint Conference on Lexical and Computational Semantics (*SEM)*.
- Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of the International Conference on Statistical Language Processing*.
- Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from Wikipedia. In *NAACL/HLT*.
- Omar F. Zaidan and Chris Callison-Burch. 2011. Crowdsourcing translation: Professional quality from non-professionals. In *Proceedings of ACL*.
- Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of ICCL*.

Cheap and easy entity evaluation

Ben Hachey Joel Nothman Will Radford

School of Information Technologies

University of Sydney

NSW 2006, Australia

ben.hachey@sydney.edu.au

{joel,wradford}@it.usyd.edu.au

Abstract

The AIDA-YAGO dataset is a popular target for whole-document entity recognition and disambiguation, despite lacking a shared evaluation tool. We review evaluation regimens in the literature while comparing the output of three approaches, and identify research opportunities. This utilises our open, accessible evaluation tool. We exemplify a new paradigm of distributed, shared evaluation, in which evaluation software and standardised, versioned system outputs are provided online.

1 Introduction

Modern entity annotation systems detect mentions in text and disambiguate them to a knowledge base (KB). Disambiguation typically returns the corresponding Wikipedia page or NIL if none exists.

Named entity linking (NEL) work is driven by the TAC shared tasks on query-driven knowledge base population (Ji and Grishman, 2011). Evaluation focuses on disambiguating queried names and clustering NIL mentions, but most systems internally perform whole-document named entity recognition, coreference, and disambiguation (Cucerzan and Sil, 2013; Pink et al., 2013; Cheng et al., 2013; Fahrni et al., 2013). *Wikification* work generally evaluates end-to-end entity annotation including KB-driven mention spotting and disambiguation (Milne and Witten, 2008b; Kulkarni et al., 2009; Ratnov et al., 2011; Ferragina and Scialla, 2010). Despite important differences in mention handling, NEL and wikification work have followed a similar trajectory. Yet to our knowledge, there are no comparative whole-document evaluations of NEL and wikification systems.

Public data sets have also driven research in whole-document entity disambiguation (Cucerzan, 2007; Milne and Witten, 2008b;

Kulkarni et al., 2009; Bentivogli et al., 2010; Hoffart et al., 2011; Meij et al., 2012). However, with many task variants and evaluation methodologies proposed, it is very difficult to synthesise a clear picture of the state of the art.

We present an evaluation suite for named entity linking, leveraging and advocating for the AIDA disambiguation annotations (Hoffart et al., 2011) over the large and widely used CoNLL NER data (Tjong Kim Sang and Meulder, 2003). This builds on recent rationalisation and benchmarking work (Cornolti et al., 2013), adding an isolated evaluation of disambiguation. Contributions include:

- a simple, open-source evaluation suite for end-to-end, whole-document NEL;
- disambiguation evaluation facilitated by gold-standard mentions;
- reference outputs from state-of-the-art NEL and wikification systems published with the suite for easy comparison;
- implementation of statistical significance and error sub-type analysis, which are often lacking in entity linking evaluation;
- a venue for publishing benchmark results continuously, complementing the annual cycle of shared tasks;
- a repository for versioned corrections to ground truth annotation.

We see this repository, at https://github.com/wikilinks/conll103_nel_eval, as a model for the future of informal shared evaluation.

We survey entity annotation tasks and evaluation, proposing a core suite of metrics for end-to-end linking and tagging, and settings that isolate mention detection and disambiguation. A comparison of state-of-the-art NEL and wikification systems illustrates how key differences in mention handling affect performance. Analysis suggests that focusing evaluation too tightly on subtasks like candidate ranking can lead to results that do not reflect end-to-end performance.

2 Tasks and metrics

The literature includes many variants of the entity annotation task and even more evaluation approaches. Systems can be invoked under two settings: given text with expressions to be linked (gold mentions); or given plain text only (system mentions). The former enables a diagnostic evaluation of disambiguation, while the latter simulates a realistic end-to-end application setting.

Within each setting, metrics may consider different subsets of the gold (\mathcal{G}) and system (\mathcal{S}) annotations. Given sets of (doc, token span, kbid) tuples, we define precision, recall and F_1 score with respect to some annotation filter f :

$$P_f = \frac{|f(\mathcal{G}) \cap f(\mathcal{S})|}{|f(\mathcal{S})|}, \quad R_f = \frac{|f(\mathcal{G}) \cap f(\mathcal{S})|}{|f(\mathcal{G})|}$$

We advocate two core metrics, corresponding to the major whole-document entity annotation tasks. *Link annotation* measures performance over every linked mention. Its filter f_L matches spans and link targets, disregarding NILs. This is particularly apt when entity annotation is a step in an information extraction pipeline. *Tag annotation* measures performance over document-level entity sets: f_T disregards span information and NILs. This is appropriate when entity annotation is used, e.g., for document indexing or social media mining (Mihalcea and Csomai, 2007; Meij et al., 2012). We proceed to ground these metrics and diagnostic variants in the literature.

2.1 End-to-end evaluation

We follow Cornolti et al. (2013) in evaluating end-to-end entity annotation, including both mention detection and disambiguation. In this context, f_L equates to Cornolti et al.’s *strong annotation match*; f_T measures what they call *entity match*.

2.2 Mention evaluation

Mention detection performance may be evaluated regardless of linking decisions. A filter f_M discards the link target (kbid). Of the present metrics, only this considers NIL-linked system mentions as different from non-mentions. For comparability with wikification, we consider an additional filter $f_{M_{KB}}$ to NEL output that retains only linked mentions. f_M and $f_{M_{KB}}$ are equivalent to Cucerzan’s (2007) *mention evaluation* and Cornolti et al.’s *strong mention match* respectively. f_M is comparable to the NER evaluation from the CoNLL

2003 shared task (Tjong Kim Sang and Meulder, 2003): span equivalence is handled the same way, but metrics here ignore mention types.

2.3 Disambiguation evaluation

Most NEL and wikification literature focuses on disambiguation, evaluating the quality of link target annotations in isolation from NER error. Providing systems with ground truth mentions makes f_L equivalent to Mihalcea and Csomai’s (2007) *sense disambiguation evaluation* and Milne and Witten’s (2008b) *disambiguation evaluation*. It differs from Kulkarni et al.’s (2009) metric in being micro-averaged (equal weight to each mention), rather than macro-averaged across documents. f_L recall is comparable to TAC’s KB *recall* (Ji and Grishman, 2011). It differs in that all mentions are evaluated rather than specific queries.

Related evaluations have also isolated disambiguation performance by: considering the links of only correctly identified mentions (Cucerzan, 2007); or only true mentions where the correct entity appears among top candidates before disambiguation (Ratinov et al., 2011; Hoffart et al., 2011; Pilz and Paass, 2012). We do not prefer this approach as it makes system comparison difficult. For comparability, we implement a filter $f_{L_{HOF}}$ that retains only Hoffart-linkable mentions having a YAGO *means* relation to the correct entity.

Tag annotation (f_T) with ground truth mentions is equivalent to Milne and Witten’s (2008b) *link evaluation*, Mihalcea and Csomai’s (2007) *keyword extraction evaluation* and Ratinov et al.’s (2011) *bag-of-titles evaluation*. It is comparable to Pilz and Paass’s (2012) *bag-of-titles evaluation*, but does not account for sequential order and keeps all gold-standard links regardless of whether they are found by candidate generation.

2.4 Further diagnostics and rank evaluation

Several evaluations in the literature are beyond the scope of this paper but planned for future versions of the code. This includes further diagnostic sub-task evaluation, particularly *candidate set recall* (Hachey et al., 2013), *NIL accuracy* (Ji and Grishman, 2011) and *weak mention matching* (Cornolti et al., 2013). With a score for each prediction, further metrics are possible: rank evaluation of tag annotation with *r-precision*, *mean reciprocal rank* and *mean average precision* (Meij et al., 2012); and rank evaluation of mentions for comparison to Hoffart et al. (2011) and Pilz and Paass (2012).

3 Data

The CoNLL-YAGO dataset (Hoffart et al., 2011) is an excellent target for end-to-end, whole-document entity annotation. It is public, free and much larger than most entity annotation data sets. It is based on the widely used NER data from the CoNLL 2003 shared task (Tjong Kim Sang and Meulder, 2003), building disambiguation on ground truth mentions. It has standard training and development splits that are representative of the held-out test data, all being sourced from the Reuters text categorisation corpus (Lewis et al., 2004), which is provided free for research purposes. Training and development comprise 1,162 stories from 22-31 August 1996 and held-out test comprises 231 stories from 6-7 December 1996. The layered annotation provides useful information for analysis including categorisation topics (e.g., general news, markets, sport) and NE type markup (PER, ORG, LOC, MISC).

The primary drawback is that KB annotations are currently present only if there is a YAGO *means* relation between the mention string and the correct entity. This means that there are a number of CoNLL entity mentions referring to entities that exist in Wikipedia that are nonetheless marked NIL in the ground truth (e.g. ‘DSE’ for ‘Dhaka Stock Exchange’). This may be addressed by using a shared repository to adopt versioned improvements to the ground truth. Annotation over CoNLL tokenisation sometimes results in strange mentions (e.g., ‘Washington-based’ instead of ‘Washington’). However, prescribed tokenisation simplifies comparison and analysis.

Another concern is that link annotation goes stale, since Wikipedia titles are only canonical with respect to a particular point in time. This is because pages may be renamed or reorganised:

- to improve editorial structure, such as downgrading an entity from having a page of its own, to a mere section in another page;
- to account for newly notable entities, such as creating a disambiguation page for a title that formerly had a single known referent; or
- because of changes in fact, such as corporate mergers and name changes.

All systems compared provide Wikipedia titles as labels, which are mapped to current titles for comparison: for each entity title t linked in the gold data, we query the Wikipedia API to find t ’s canonical form t_c and retrieve titles of all redirects to t_c .

4 Reference systems

Even on public data sets, comparison to published results can be very difficult and extremely costly (Fokkens et al., 2013). We include reference system output in our repository for simple comparison. Other researchers are welcome to add reference output, providing a continuous benchmark that complements the annual cycle of large shared tasks like TAC KBP.

4.1 TagMe

TagMe (Ferragina and Scaiella, 2010) is an end-to-end wikification system specialising in short texts. TagMe performs best among publicly available wikification systems (Cornolti et al., 2013). Mention detection uses a dictionary of anchor text from links between Wikipedia pages. Candidate ranking is based on entity relatedness (Milne and Witten, 2008a), followed by mention pruning. We use thresholds on annotation scores supplied by Marco Cornolti (personal communication) of 0.289 and 0.336 respectively for mention/link and tag evaluation. TagMe annotations may not align with CoNLL token boundaries, e.g., `<annot title=“Oakland, New Jersey”>OAKLAND, N.J.</annot>`. Before evaluation, we extend annotations to overlapping tokens.

4.2 AIDA

AIDA (Hoffart et al., 2011) is the system presented with the CoNLL-YAGO dataset and places emphasis on state-of-the-art ranking of candidate entity sets. Mentions are ground truth from the CoNLL data to isolate ranking performance, equivalent to applying the $f_{L_{\text{HOF}}}$ filter. Ranking is informed by a graph model of entity compatibility.

4.3 Schwa

Schwa (Radford et al., 2012) is a heuristic NEL system based on a TAC 2012 shared task entrant. Mention detection uses a NER model trained on news text followed by rule-based coreference. Disambiguation uses an unweighted combination of KB statistics, document compatibility (Cucerzan, 2007), graph similarity and targeted textual similarity. Candidates that score below a threshold learned from TAC data are linked to NIL. The system is very competitive, performing at 93% and 97% respectively of the best accuracy numbers we know of on 2011 and 2012 TAC evaluation data (Cucerzan and Sil, 2013).

System	Mentions	Filter	P	R	F_1
Cucerzan	System	f_M	82.2	84.8	83.5
Schwa	System	f_M	86.9	76.7	81.5
TagMe	System	$f_{M_{KB}}$	75.2	60.4	67.0
Schwa	System	$f_{M_{KB}}$	82.5	74.5	78.3

Table 1: Mention detection results. Cucerzan results as reported (Cucerzan, 2007).

5 Results

We briefly report results over the reference systems to highlight characteristics of the evaluation metrics and task settings. Results hinge upon Schwa since we have obtained only its output in all settings. Except where noted, all differences are significant ($p < 0.05$) according to approximate randomisation (Noreen, 1989), permuting annotations over whole documents.

5.1 Mention evaluation

Table 1 evaluates mentions with and without NILs. None of the systems reported use a CoNLL-trained NER tagger, for which top shared task participants approached 90% F_1 in a stricter evaluation than f_M . We note the impressive numbers reported by Cucerzan (2007) using a novel approach to mention detection based on capitalisation and corpus co-occurrence statistics, and the similar performance¹ to Schwa, whose NER component is trained on another news corpus.

In wikification, NIL-linked mentions may not be relevant, and it may suffice to identify only the most canonical forms of names, rather than all mentions in a coreference chain. With $f_{M_{KB}}$, Schwa has much higher recall than TagMe, though TagMe’s precision is understated because it generates non-NE annotations that are not present in the CoNLL-YAGO ground truth (e.g., linking ‘striker’ to Forward (association football)).

5.2 Disambiguation evaluation

Table 2 contains results isolating disambiguation performance. AIDA ranking outperforms Schwa according to both the link ($f_{L_{HOF}}$) and tag metrics ($f_{T_{HOF}}$). If we remove the Hoffart et al. (2011) linkable constraint, we observe that Schwa disambiguation performance loses about 8 points in precision on the link metric (f_L) and 2 points on the tag metric (f_T). This suggests that disambiguation

¹Significance cannot be tested since we do not have the Cucerzan (2007) output.

System	Mentions	Filter	P	R	F_1
Schwa	Gold	f_L	67.5	78.3	72.5
Schwa	Gold	$f_{L_{HOF}}$	79.7	78.3	79.0
AIDA	Gold	$f_{L_{HOF}}$	83.2	83.2	83.2
Schwa	Gold	f_T	77.8	77.7	77.7
Schwa	Gold	$f_{T_{HOF}}$	80.1	77.6	78.8
AIDA	Gold	$f_{T_{HOF}}$	87.7	84.2	85.9

Table 2: Disambiguation results for mention-level linking and document-level tagging.

System	Mentions	Filter	P	R	F_1
TagMe	System	f_L	63.2	50.7	56.3
Schwa	System	f_L	67.6	61.0	64.2
TagMe	System	f_T	65.0	65.4	65.2
Schwa	System	f_T	71.2	62.6	66.6

Table 3: End-to-end results for mention-level linking and document-level tagging.

evaluation without the linkable constraint is important, especially if the application requires detecting and disambiguating all mentions.

The comparison here highlights a notable evaluation intricacy. The Schwa system disambiguates all gold mentions rather than those with KB links, and the document compatibility approach means that evidence from a NIL mention may offer confounding evidence when linking linkable mentions. Further, although using the same mentions, systems use search resources with different recall characteristics, so the Schwa system may not retrieve the correct candidate to disambiguate.

5.3 End-to-end evaluation

Finally, Table 3 contains end-to-end entity annotation results. Again, these results highlight key differences in mention handling between NEL and wikification. Coreference modelling helps NEL detect and link ambiguous names (e.g., ‘President Bush’) that refer to the same entity as unambiguous names in the same text (e.g., ‘George W. Bush’). And restricting the the universe to named entities is appropriate for the CoNLL-YAGO data. The advantage is marked in the mention-level link evaluation (f_L). However, the systems are statistically indistinguishable in the document-level tag evaluation (f_T). Thus the extra NER and coreference machinery may not be justified if the application is document indexing or social media mining (Meij et al., 2012), wherein a KB-driven mention detector may be favourable for other reasons.

Error	$f_{L_{\text{HOF}}}$		f_L	
	AIDA	Schwa	TagMe	Schwa
wrong link	752	896	429	605
link as nil	-	79	-	111
nil as link	-	-	183	337
missing	-	-	1,780	1,031
extra	-	-	1,663	927

Table 4: $f_{L_{\text{HOF}}}$ and f_L error profiles.

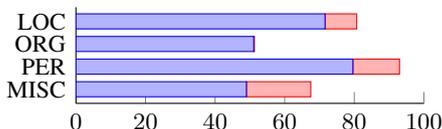


Figure 1: Schwa f_L and $f_{L_{\text{HOF}}}$ F_1 for NE types

6 Analysis

We analyse the types of error that a system makes. We also harness the multi-layered annotation to quantify the effect of NE type and document topic.

By error type Table 4 shows error counts based on the disambiguation link evaluation with the linkable constraint ($f_{L_{\text{HOF}}}$) and the end-to-end link evaluation (f_L). Errors are divided as follows:

wrong link: mention linked to wrong KB entry

link as nil: KB-entity mention linked to NIL

nil as link: NIL mention linked to the KB

missing: true mention not detected

extra: mention detected spuriously

AIDA outperforms Schwa under the linkable evaluation, making fewer wrong link errors. Schwa also overgenerates NIL, which may reflect candidate recall errors or a conservative disambiguation threshold. On the end-to-end evaluation, Schwa makes more linking errors (wrong link, link as nil, nil as link) than TagMe, but fewer in mention detection, leading to higher overall performance.

By entity type Figure 1 evaluates only mentions where the CoNLL 2003 corpus (Tjong Kim Sang and Meulder, 2003) marks a NE mention of each type. This is based on the link evaluation of Schwa. The left and right bars correspond to end-to-end (f_L) and disambiguation ($f_{L_{\text{HOF}}}$) F_1 respectively. In accord with TAC results (Ji and Grishman, 2011), high accuracy can be achieved on PER when a full name is given, while ORG is substantially more challenging. MISC entities are somewhat difficult to disambiguate, with identification errors hampering end-to-end performance.

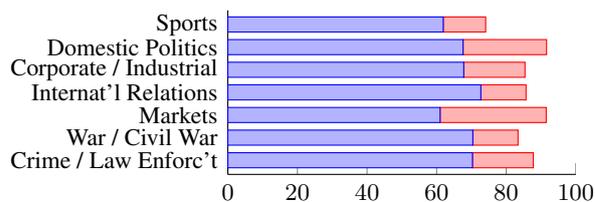


Figure 2: Schwa f_L and $f_{L_{\text{HOF}}}$ F_1 for top topics

By topical category The underlying Reuters Corpus documents are labelled with topic, country and industry codes (Lewis et al., 2004). Figure 2 reports F_1 on test documents from each frequent topic. It highlights that much ambiguity remains unresolved in *Sports*, while very high performance linking is attainable in categories such as *Markets* and *Domestic Politics*, only when given ground truth linkable mentions.

7 Conclusion

We surveyed entity annotation tasks and advocated a core set of metrics for mention, disambiguation and end-to-end evaluation. This enabled a direct comparison of state-of-the-art NEL and wikification systems, highlighting the effect of key differences. In particular, NER and coreference modules make NEL approaches suitable for applications that require all mentions, including ambiguous names and entities that are not in the KB. For applications where document-level entity tags are appropriate, the NEL and wikification approaches we evaluate have similar performance.

The big picture we wish to convey is a new approach to community evaluation that makes benchmarking and qualitative comparison cheap and easy. In addition to the code being open source, we use the repository to store reference system output, and – we hope – emendations to the ground truth. We encourage other researchers to contribute reference output and hope that this will provide a continuous benchmark to complement the current cycle of shared tasks.

Acknowledgements

Many thanks to Johannes Hoffart, Marco Cornolti, Xiao Ling and Edgar Meij for reference outputs and guidance. Ben Hachey is the recipient of an Australian Research Council Discovery Early Career Researcher Award (DE120102900). The other authors were supported by the Capital Markets CRC Computable News project.

References

- Luisa Bentivogli, Pamela Forner, Claudio Giuliano, Alessandro Marchetti, Emanuele Pianta, and Kateryna Tymoshenko. 2010. Extending English ACE 2005 corpus annotation with ground-truth links to Wikipedia. In *COLING Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, pages 19–27.
- Xiao Cheng, Bingling Chen, Rajhans Samdani, Kai-Wei Chang, Zhiye Fei, Mark Sammons, John Wieting, Subhro Roy, Chizheng Wang, and Dan Roth. 2013. Illinois cognitive computation group UI-CCG TAC 2013 entity linking and slot filler validation systems. In *Text Analysis Conference*.
- Marco Cornolti, Paolo Ferragina, and Massimiliano Ciaramita. 2013. A framework for benchmarking entity-annotation systems. In *22nd International Conference on the World Wide Web*, pages 249–260.
- Silviu Cucerzan and Avirup Sil. 2013. The MSR systems for entity linking and temporal slot filling at TAC 2013. In *Text Analysis Conference*.
- Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 708–716.
- Angela Fahrni, Benjamin Heinzerling, Thierry Göckel, and Michael Strube. 2013. HITS' monolingual and cross-lingual entity linking system at TAC 2013. In *Text Analysis Conference*.
- Paolo Ferragina and Ugo Scaiella. 2010. TAGME: On-the-fly annotation of short text fragments (by Wikipedia entities). In *19th International Conference on Information and Knowledge Management*, pages 1625–1628.
- Antske Fokkens, Marieke van Erp, Marten Postma, Ted Pedersen, Piek Vossen, and Nuno Freire. 2013. Offspring from reproduction problems: What replication failure teaches us. In *51st Annual Meeting of the Association for Computational Linguistics*, pages 1691–1701.
- Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R. Curran. 2013. Evaluating entity linking with Wikipedia. *Artificial Intelligence*, 194:130–150.
- 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 *Conference on Empirical Methods in Natural Language Processing*, pages 782–792.
- Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *49th Annual Meeting of the Association for Computational Linguistics*, pages 1148–1158.
- Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of Wikipedia entities in web text. In *15th International Conference on Knowledge Discovery and Data Mining*, pages 457–466.
- David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397.
- Edgar Meij, Wouter Weerkamp, and Maarten de Rijke. 2012. Adding semantics to microblog posts. In *5th International Conference on Web Search and Data Mining*, pages 563–572.
- Rada Mihalcea and Andras Csomai. 2007. Wikify! Linking documents to encyclopedic knowledge. In *16th Conference on Information and Knowledge Management*, pages 233–242.
- David Milne and Ian H. Witten. 2008a. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *AAAI Workshop on Wikipedia and Artificial Intelligence*, pages 25–30.
- David Milne and Ian H. Witten. 2008b. Learning to link with Wikipedia. In *17th Conference on Information and Knowledge Management*, pages 509–518.
- Eric W. Noreen. 1989. *Computer Intensive Methods for Testing Hypotheses*. John Wiley & Sons.
- Anja Pilz and Gerhard Paass. 2012. Collective search for concept disambiguation. In *24th International Conference on Computational Linguistics*, pages 2243–2258.
- Glen Pink, Will Radford, Will Cannings, Andrew Naoum, Joel Nothman, Daniel Tse, and James R. Curran. 2013. SYDNEY_CMCRC at TAC 2013. In *Text Analysis Conference*.
- Will Radford, Will Cannings, Andrew Naoum, Joel Nothman, Glen Pink, Daniel Tse, and James R. Curran. 2012. (Almost) Total Recall – SYDNEY_CMCRC at TAC 2012. In *Text Analysis Conference*.
- Lev Ratnov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to Wikipedia. In *49th Annual Meeting of the Association for Computational Linguistics*, pages 1375–1384.
- Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Conference On Computational Natural Language Learning*, pages 142–147.

Identifying Real-Life Complex Task Names with Task-Intrinsic Entities from Microblogs

Ting-Xuan Wang* and Kun-Yu Tsai and Wen-Hsiang Lu

National Cheng Kung University
Tainan, Taiwan

{P78981320, P76014460, whlu}@mail.ncku.edu.tw

Abstract

Recently, users who search on the web are targeting to more complex tasks due to the explosive growth of web usage. To accomplish a complex task, users may need to obtain information of various entities. For example, a user who wants to travel to Beijing, should book a flight, reserve a hotel room, and survey a Beijing map. A complex task thus needs to submit several queries in order to seeking each of entities. Understanding complex tasks can allow a search engine to suggest related entities and help users explicitly assign their ongoing tasks.

1 Introduction

The requirement of searching for complex tasks dramatically increases in current web search. Users not always search for single information need (Liao et al., 2012). To accomplish a real-life complex task, users usually need to obtain various information of distinct entities on the web. In this paper, we define the necessary entities for a complex task as task-intrinsic entities. For example, a complex task “travel to Beijing” has at least three task-intrinsic entities, including a flight ticket, hotel room, and maps. Therefore, users need submit several queries in order to seek all of the necessary entities. However, conventional search engines are careless of latent complex tasks behind a search query. Users are guided to search for each task-intrinsic entity one by one to accomplish their complex task inefficiently.

Figure 1 shows a complex task consisting of a task name “travel to Beijing” and several task-intrinsic entities. A task name is composed of a task event and a task topic. The task event triggers users to perform exploratory or comparative search behaviors such as “prepare

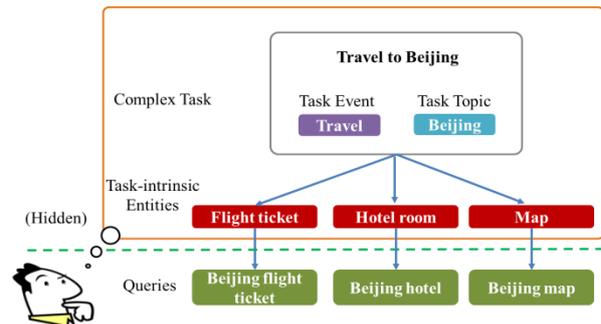


Figure 1. The structure of a complex task with task-intrinsic entities and related queries.

something”, “buy something” or “travel to somewhere”. The task topic is the subject of interest in the complex task. Task-intrinsic entities are intrinsically demanded by the complex task. The three queries “Beijing flight ticket”, “Beijing hotel”, and “Beijing map” are driven by the information need of each of task-intrinsic entities with topic “Beijing” and event “travel” for the hidden complex task “travel to Beijing”.

According to our observation, users may describe details of a complex task to be done or already completed via microblogs, e.g., Twitter or Weibo¹. Microblogs are a miniature version of traditional weblogs. In recent years, many users post and share their life details with others on microblogs every day. Due to the post length limitation (only 140 characters in case of Weibo), users tend to only describe key points. Table 1 shows an example of a microblog. We can find that the user, who has an ongoing complex task “北京旅遊(travel to Beijing)”, mentioned two task-intrinsic entities “機票 (flight ticket)” and “飯店 (hotel)”.

In this work, we address the problem of how to help users efficiently accomplish a complex task when submitting a single query or multiple queries.

¹ Weibo: <http://weibo.com>

Chinese
今天已經訂好機票，只剩下找間飯店，就等著下禮拜去北京旅遊了~好期待!
English Translation
I have already booked a flight today, and I only have to find a hotel . I'm about to travel to Beijing next week - good anticipation!

Table 1. A microblog post from Weibo mentioning an ongoing complex task “北京旅遊 (travel to Beijing)”

We divide the problem into the following three major sub-problems.

1. Find task-intrinsic entities for the complex task.
2. Generate a task name for the complex task.
3. Suggest proper search results covering all desired entities for the complex task.

The above three problems are very important but non-trivial to solve. In this preliminary work, we only focus on first two sub-problems. We proposed an entity-driven complex task model (ECTM) to automatically generate complex task names and related task-intrinsic entities. To evaluate our proposed ECTM, we conducted experiments on a large dataset of real-world query logs. The experimental results show that our ECTM is able to identify a comprehensive complex task name with the task-intrinsic entities and help users accomplish the complex task with less effort.

2 Related Work

Recent studies show that about 75% of search sessions searching for complex tasks (Feild and Allan, 2013). To help users deal with their complex search tasks, researchers devoted their efforts to understand and identify complex tasks from search sessions. Boldi et al. (2002) proposed a graph-based approach to dividing a long-term search session into search tasks. Guo and Agichtein (2010) made the attempt to investigate the hierarchical structure of a complex task with a series of search actions based on search sessions. Cui et al. (2011) proposed random walk based methods to discover search tasks from search sessions. Kotov et al. (2011) noticed that a multi-goal task may require a user to issue a series of queries, spanning a long period of time and multiple search sessions. Thus, they addressed the problem of modeling and analyzing complex cross-session search tasks. Lucchese et al. (2011) tried to identify task-based sessions in query logs by semantic-based features extracted

from Wiktionary and Wikipedia to overcome lack of semantic information. Ji et al. (2011) proposed a graph-based regularization algorithm to predict popular search tasks and simultaneously classify queries and web pages by building two content-based classifiers. White et al. (2013) improved the traditional personalization methods for search-result re-ranking by exploiting similar tasks from other users to re-rank search results. Wang et al. (2013) addressed the problem of extracting cross session tasks and proposed a task partition algorithm based on several pairwise similarity features. Raman et al. (2013) investigated intrinsic diversity (ID) for a search task and proposed a re-ranking algorithm according to the ID tasks.

A complex task consists of several sub-tasks, and each sub-task goal may be composed of a sequence of search queries. Therefore, modeling the sub-tasks is necessary for identifying a complex task. Klinkner (2008) proposed a classification-based method to divide a single search session into tasks and sub-tasks based on the four types of features, including time, word, query log sequence, and web search. Lin et al. (2012) defined a search goal as an action-entity pair and utilized web trigram to generate fine-grained search goals. Agichtein et al. (2012) conducted a comprehensive analysis of search tasks and classified them based on several aspects, such as intent, motivation, complexity, work-or-fun, time-sensitive, and continued-or-not. Jones and Yamamoto et al. (2012) proposed an approach to mining sub-tasks for a task using query clustering based on bid phrases provided by advertisers. The most important difference between our work and previous works is that we further try to generate task names with related task-intrinsic entities. To the best of our knowledge, there is no existing approach to utilizing microblogs in dealing with task identification and generating human-interpretable names.

3 Entity-driven Complex Task Model

3.1 Problem Formulation

Given a query q , we aim to identify the complex task for the query. Since the single query is not able to describe a complex task. Our proposed ECTM model introduces an expanded query set \mathbf{Q}_t for helping identify the task t . Thus, $P(t|q)$ can be formulated as follows:

$$P(t|q) = \sum_{\mathbf{Q}_t} P(\mathbf{Q}_t|q)P(t|\mathbf{Q}_t, q) \quad (1)$$

Since the expanded query set \mathbf{Q}_t always contain

the input query q , the Equation (1) can thus be approximated as:

$$P(t|q) = \sum_{\mathbf{Q}_t} P(\mathbf{Q}_t|q)P(t|\mathbf{Q}_t), \quad (2)$$

where $P(\mathbf{Q}_t|q)$ is the query expansion model. For $P(t|\mathbf{Q}_t)$ we utilize a set of microblog posts \mathbf{m} for identifying the complex task t and obtain the following equation:

$$P(t|\mathbf{Q}_t) = \sum_{\mathbf{m}} P(\mathbf{m}|\mathbf{Q}_t)P(t|\mathbf{m}, \mathbf{Q}_t). \quad (3)$$

For $P(t|\mathbf{m}, \mathbf{Q}_t)$ in Equation (3), the query set \mathbf{Q}_t can be omitted since the microblog post set \mathbf{m} contains \mathbf{Q}_t . The Equation (3) can thus be modified as follows:

$$P(t|\mathbf{Q}_t) = \sum_{\mathbf{m}} P(\mathbf{m}|\mathbf{Q}_t)P(t|\mathbf{m}). \quad (4)$$

Finally, the ECTM can be obtained as follows:

$$P(t|q) = \sum_{\mathbf{Q}_t} P(\mathbf{Q}_t|q) \sum_{\mathbf{m}} P(\mathbf{m}|\mathbf{Q}_t)P(t|\mathbf{m}), \quad (5)$$

where $P(\mathbf{Q}_t|q)$ is the query expansion model, $P(\mathbf{m}|\mathbf{Q}_t)$ is microblog retrieval model, and $P(t|\mathbf{m})$ is task identification model. In the following section, we will describe the three models in detail respectively.

3.2 Query Expansion Model

In fact, only using a single query is insufficient to identify the latent complex task. We thus try to extract task-coherent queries from search sessions. According to our observation, users may persistently search for the same complex task in a period of time. However, users may also simultaneously interleave search for multiple different tasks (MacKay and Watters, 2008; Liu and Belkin, 2010). Therefore, identifying task-coherent queries from search sessions is an important issue. We perform the following processes in order to extract task-coherent queries.

Given a query log and an input query q , we first separate queries in the log into search sessions with the time gap of 30 minutes. We extract search sessions containing the input query q and thus obtain a set of sessions \mathcal{S}_q . To extract task-coherent queries \mathbf{Q}_t from the session set \mathcal{S}_q , we employ log-linear model (LLM) with the following three useful features:

Average Query Frequency: In most cases, the frequency of queries can reflect their importance. To avoid a long session resulting in high query frequency, we calculate the normalized query frequency as:

$$f_{AQFrequency}(q_t) = \frac{1}{|\mathcal{S}_{q_t}|} \times \sum_{s \in \mathcal{S}_{q_t}} \frac{freq(q_t, s)}{|s|}, \quad (6)$$

where $freq(q_t, s)$ is the frequency of the query q_t in session s , \mathcal{S}_{q_t} is the sessions containing q_t ,

$|s|$ is the number of queries in session s , and $|\mathcal{S}_{q_t}|$ is the number of sessions containing query q_t in the set \mathcal{S}_{q_t} .

Session Coverage: The queries occurring in several sessions are possible candidates in terms of task-coherence. In order to favor queries occurring in many sessions, we use average session frequency, which can be calculated as follows:

$$f_{ASFrequency}(q_t) = \exp\left(\frac{|\mathcal{S}_{q_t}|}{|\mathcal{S}_q|}\right), \quad (7)$$

where $|\mathcal{S}_q|$ is the number of sessions containing the input query q in the set \mathcal{S}_q , $|\mathcal{S}_{q_t}|$ is the number of sessions containing query q_t in the set \mathcal{S}_{q_t} , and $\exp(\cdot)$ is the exponential function.

Average Query Distance: Since queries which close to the input query in a search session may have high task-coherence for the latent complex task. We thus use normal distribution to estimate the task-coherence for each query:

$$f_{AQDistance}(q_t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{d^2}{2\sigma^2}}, \quad (8)$$

where σ is standard deviation (is empirically set 0.2 in this work), d is the average number of queries between q_t and input query q in sessions.

We employ log-linear model to calculate the probability of each candidate task-coherent query based on the features described above:

$$P(q_t; \mathbf{W}) = \frac{\exp(\sum_{i=1}^{|F|} w_i f_i(q_t))}{Z(\mathbf{Q}_t)}, \quad (9)$$

where \mathbf{Q}_t is the set of all candidate queries in the session set \mathcal{S}_q , $|F|$ is the number of used feature functions $f_i(q_t)$, \mathbf{W} is the set of weighting parameters w_i of feature functions, and $Z(\mathbf{Q}_t)$ is a normalizing factor set to the value $Z(\mathbf{Q}_t) = \sum_{q_t \in \mathbf{Q}_t} \exp(\sum_{i=1}^{|F|} w_i f_i(q_t))$.

3.3 Microblog Retrieval Model

Since the task names are not always observable in the expanded query set \mathbf{Q}_t , we thus need further expanding \mathbf{Q}_t by retrieving microblog posts. The basic idea is that a microblog post containing all queries in \mathbf{Q}_t may also contain the task name (see the example in Table 1). In fact, the queries in the query set \mathbf{Q}_t usually consist of a topic name and a task-intrinsic entity. For example a query “北京機票 (Beijing flight ticket)” contains a topic “北京 (Beijing)” and an entity “機票 (flight ticket)”. Therefore, we first try to extract task-intrinsic entities from the query set \mathbf{Q}_t by extracting all common nouns in each of queries. We can thus obtain a list of task-intrinsic

entities E_t ordered by the occurrence frequency of each entity. Since a microblog post may only contain a part of entities for a complex task, we generate pseudo queries based on all subsets containing two or three entities from top- n entities of E_t . Finally, we use all generated pseudo queries to retrieve microblog posts.

3.4 Task Identification Model

To identify a suitable task name from retrieved microblog posts, there are two steps in this model, including candidate task name extraction and correct task name determination.

Candidate Task Name Extraction

For each retrieved microblog post, we first extract all bigrams and trigrams which match the POS (part of speech) patterns listed in Table 2. According to our observation, the POS of a task topic is usually a proper noun (N_p) and the POS of a task event is usually a transitive verb (V_t) + common noun (N_c) or an intransitive verb (V_i). On the other hand, a task topic may be the most important term in related search sessions S . More specifically, the term with the POS of proper noun and the highest occurrence count in the Q_t . We thus consider the term as a candidate topic (notated as $\langle T \rangle$) and adopt two related task POS patterns, i.e., $V_t + \langle T \rangle + N_c$ and $\langle T \rangle + V_i$.

Topic POS	Event POS	Task POS Pattern
N_p	$V_t + N_c$	$V_t + N_p + N_c$
	V_i	$N_p + V_i$
$\langle T \rangle$	$V_t + N_c$	$V_t + \langle T \rangle + N_c$
	V_i	$\langle T \rangle + V_i$

Table 2. Adopted POS patterns for extracting candidate task names from microblog posts.

Correct Task Name Determination

Different from long-text documents (e.g., webpages), microblog posts are relatively short and hard to find features based on special sections in content (e.g., anchor text, title, or blocks). Therefore, we use five efficient features proposed by Zeng et al. (2004) to extract complex task names from short-text snippets, such as microblog post or search-result snippets. The features proposed by Zeng et al. including TFIDF, phrase length, intra-cluster similarity, cluster entropy, and phrase independence. Furthermore, in this work, we plus two practical features *task name coverage* (the percentage of microblog posts containing the candidate task name) and *chi-square score* (Manning, 1999).

Based on the set of extracted candidate task names T_q for the input query q , we also utilized LLM to select the potential task names with the highest likelihood. The LLM for identifying complex task names is given as follows:

$$P(t; \Gamma) = \frac{\exp(\sum_{j=1}^{|K|} \gamma_j k_j(t))}{Z(T_q)}, \quad (10)$$

where Γ is the set of weighting parameters γ_j of feature functions $k_j(t)$, $|K|$ is the number of feature functions $k_j(t)$, $Z(T_q)$ is a normalizing factor set to $\sum_{t \in T_q} \exp(\sum_{j=1}^{|K|} \gamma_j k_j(t))$.

4 Experiments

4.1 Data

We use a one-month query logs from the Sogou search engine, which contains 21,422,773 records and 3,163,170 distinct queries. Each record contains user ID, query, clicked URL, user clicked order for the query, and the search-result rank of the clicked URL. We group query records into sessions according to user ID. Since a complex search task may take a long time to accomplish, we used one week as the time gap to split sessions, and finally obtained 264,360 sessions. For microblogs, we collected the top 50 posts for each pseudo query from Weibo.

To evaluate the performance of our proposed ECTM model, we manually selected 30 testing queries from sessions which are searching for complex tasks. For each query, we employ three annotators to label complex task names. Three annotators independently annotated 30 queries. We further examined the labeled results, and unified the similar task names. For instances, “北京旅遊 (travel to Beijing)” and “北京旅行 (trip to Beijing)” were be unified to “北京旅遊 (travel to Beijing)”. Table 3 shows an example of testing query with labeled task name and task-intrinsic entities.

Query	Labeled Task Name	Labeled Task-Intrinsic Entities
Chinese		
北京旅行社	北京旅遊	地圖, 天氣, 飯店 機票, 行程表
English Translation		
Beijing travel agency	travel to Beijing	map, weather, hotel, flight tickets, schedule

Table 3. An example query “北京旅行社 (Beijing travel agency)” with labeled task name and task-intrinsic entities.

4.2 Compared Methods

We compare our approach with the state-of-the-art phrase extraction approach from short-text snippet (e.g., microblog posts or search result snippets):

- **Cluster_Q_RS (baseline)**: The method is proposed by Zeng et al. (2004), which try to identify important phrases from search result snippets. They proposed five features including TFIDF, phrase length, intra-cluster similarity, cluster entropy, and phrase independence.
- **Cluster_EQ_RS**: Since the above method only aim to identify important phrases from a single query, the result should be not fair for the problem addressed in this work. We try to enhance Cluster_Q_RS using expanded search-result snippets proposed in this work.
- **ECTM_RS**: This method further use our suggested POS patterns for extracting candidate task names and use all features proposed in Section 3.4.2.
- **ECTM_MB**: The only difference between this method and the above method is that the method try to identify task names from microblog posts.

4.3 Parameter Selection

The weights of feature functions are learned by five-fold cross-validation based on our labeled data. We use the same weights for the all of following experiments. Furthermore, determining the number of task-intrinsic entities used in generating pseudo queries is most critical in this work. We show the top n average coverage rate and average precision of extracted entities for our 30 testing queries in Figure 2.

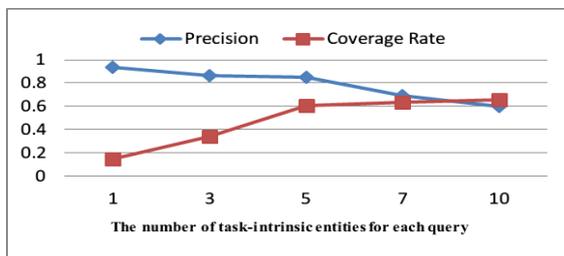


Figure 2. The precision and coverage rate of top n entities used in our microblog retrieval model

We found that using top 5 task-intrinsic entities can achieve the best results. Therefore, for each query, we will generate 20 (i.e., $C_2^5 + C_3^5$) pseudo queries and we retrieved top 10 microblog posts for each pseudo queries (totally 200 posts for each testing query).

4.4 Results of Task Name Identification

We use average top k inclusion rate as the metrics. For a set of queries, its top k inclusion rate is defined as the percentage of the query set whose correct task names occur in the first k identified task names. The overall results are shown in Table 4. We can see that our ECTM_MB outperform other methods. The ECTM_MB can identify correct task names within the first three recommendations. Unsurprisingly, Cluster_Q_RS achieved worst inclusion rate. The reason is that Cluster_Q_RS try to find comprehensive complex task name based on search results from only a single query. Most of task names suggested by Cluster_Q_RS are simple task names i.e., the sub-tasks for the latent complex task, such as “預訂機票 (book flight tickets)”. For ECTM_RS, which is a variation of Cluster_EQ_RS, it achieved slightly better performance by adding the restrictions of POS patterns for extracting candidate task names. Since some identified task names in Cluster_EQ_RS may not semantically suitable, ECTM_RS’s approach can efficiently deal with this problem. Furthermore, we also found that using search-result snippets may generate worse task names than using microblog posts. According to our investigating on the two types of the short-text-snippet resources, the search-result snippets are very diverse and task-extrinsic while microblog posts are task-coherent in describing real-life tasks.

Top k inclusion rate	Top1	Top3	Top5	Top10
Cluster_Q_RS	0.28	0.33	0.37	0.47
Cluster_EQ_RS	0.40	0.43	0.50	0.73
ECTM_RS	0.43	0.43	0.57	0.83
ECTM_MB	0.87	1	1	1

Table 4. The results of compared methods

5 Conclusion

In this work, we proposed an entity-driven complex task model (ECTM), which addressed the problem of improving user experience when searching for a complex task. Experimental results show that ECTM efficiently identifies complex tasks with various task-intrinsic entities. Nevertheless, there are still some problems that need to be solved. In the future, we will try to investigate ranking algorithms for developing a novel complex-task-based search engine, which can deal with queries based on complex tasks in real life.

References

- Agichtein, E., White, R. W., Dumais, S. T., and Bennett, P. N. Search, Interrupted: Understanding and Predicting Search Task Continuation. In *Proc. of SIGIR*, 2012.
- Boldi, P., Bonchi, F., Castillo, C., Donato, D., Gionis, A., and Vigna, S. The Query-Flow Graph: Model and Applications. In *Proc. of CIKM*, 2008.
- Cui, J., Liu, H., Yan, J., Ji L., Jin R., He, J., Gu, Y., Chen, Z., and Du, X. Multi-view Random Walk Framework for Search Task Discovery from Click-through Log. In *Proc. of CIKM*, 2011.
- Feild, H. and Allan, J. Task-Aware Query Recommendation. In *Proc. of SIGIR*, 2013.
- Guo, Q. and Agichtein, E. Ready to Buy or Just Browsing? Detecting Web Searcher Goals from Interaction Data. In *Proc. of SIGIR*, 2010.
- Ji, M., Yan, J., Gu, S., Han, J., He, X., Zhang, W. V., and Chen, Z. Learning Search Tasks in Queries and Web Pages via Graph Regularization. In *Proc. of SIGIR*, 2011.
- Jones, R., and Klinkner, K. Beyond the Session Timeout: Automatic Hierarchical Segmentation of Search Topics in Query Logs. In *Proc. of CIKM*, 2008.
- Kotov, A., Bennett, P. N., White, R. W., Dumais, S. T., and Teevan, J. Modeling and Analysis of Cross-Session Search Tasks. In *Proc. of SIGIR*, 2011.
- Liao, Z., Song, Y., He, L.-W., and Huang, Y. Evaluating the Effectiveness of Search Task Trails. In *Proc. of WWW*, 2012.
- Lin, T., Pantel, P., Gamon, M., Kannan, A., and Fuxman, A. Active Objects: Actions for Entity-Centric Search. In *Proc. of WWW*, 2012.
- Liu, J. and Belkin, N. J. Personalizing Information Retrieval for Multi-Session Tasks: The Roles of Task Stage and Task Type. In *Proc. of SIGIR*, 2010.
- Lucchese, C., Orlando, S., Perego, R., Silvestri, F., and Tolomei, G. Identifying Task-based Sessions in Search Engine Query Logs. In *Proc. of WSDM*, 2011.
- MacKay, B. and Watters, C. Exploring Multi-Session Web Tasks. In *Proc. of CHI*, 2008.
- Manning, C. D., Schütze, H. Foundations of Statistical Natural Language Processing. The MIT Press. Cambridge, US, 1999.
- Raman, K., Bennett, P. N., and Collins-Thompson, K. Toward Whole-Session Relevance: Exploring Intrinsic Diversity in Web Search. In *Proc. of SIGIR*, 2013.
- Wang, H., Song, Y., Chang, M.-W., He, X., White, R. W., and Chu, W. Learning to Extract Cross-Session Search Tasks. In *Proc. of WWW*, 2013.
- White, R. W., Chu, W., Hassan, A., He, X., Song, Y., and Wang, H. Enhancing Personalized Search by Mining and Modeling Task Behavior. In *Proc. of WWW*, 2013.
- Yamamoto, T., Sakai, T., Iwata, M., Yu, C., Wen, J.-R., and Tanaka, K. The Wisdom of Advertisers: Mining Subgoals via Query Clustering. In *Proc. of CIKM*, 2012.
- Zeng, H.-J., He, Q.-C., Chen, Z., Ma, W.-Y., and Ma, J. Learning to Cluster Web Search Results. In *Proc. of SIGIR*, 2004.

Mutual Disambiguation for Entity Linking

Eric Charton

Polytechnique Montréal
Montréal, QC, Canada

eric.charton@polymtl.ca

Ludovic Jean-Louis

Polytechnique Montréal

ludovic.jean-louis@polymtl.ca

Marie-Jean Meurs

Concordia University
Montréal, QC, Canada

marie-jean.meurs@concordia.ca

Michel Gagnon

Polytechnique Montréal

michel.gagnon@polymtl.ca

Abstract

The disambiguation algorithm presented in this paper is implemented in SemLinker, an entity linking system. First, named entities are linked to candidate Wikipedia pages by a generic annotation engine. Then, the algorithm re-ranks candidate links according to mutual relations between all the named entities found in the document. The evaluation is based on experiments conducted on the test corpus of the TAC-KBP 2012 entity linking task.

1 Introduction

The Entity Linking (EL) task consists in linking name *mentions* of named entities (NEs) found in a document to their corresponding entities in a reference Knowledge Base (KB). These NEs can be of type person (PER), organization (ORG), etc., and they are usually represented in the KB by a Uniform Resource Identifier (URI). Dealing with ambiguity is one of the key difficulties in this task, since mentions are often highly polysemous, and potentially related to many different KB entries. Various approaches have been proposed to solve the named entity disambiguation (NED) problem. Most of them involve the use of surface forms extracted from Wikipedia. Surface forms consist of a word or a group of words that match lexical units like *Paris* or *New York City*. They are used as matching sequences to locate corresponding candidate entries in the KB, and then to disambiguate those candidates using similarity measures.

The NED problem is related to the *Word Sense Disambiguation* (WSD) problem (Navigli, 2009), and is often more challenging since mentions of NEs can be highly ambiguous. For instance, names of places can be very common as is Paris, which refers to 26 different places in Wikipedia. Hence, systems that attempt to address the NED

problem must include disambiguation resources. In the context of the Named Entity Recognition (NER) task, such resources can be generic and generative. This generative approach does not apply to the EL task where each entity to be linked to a semantic description has a specific word context, marker of its exact identity.

One of the classical approach to conduct the disambiguation process in NED applications is to consider the context of the mention to be mapped, and compare this context with contextual information about the potential target entities (see for instance the KIM system (Popov et al., 2003)). This is usually done using similarity measures (such as cosine similarity, weighted Jaccard distance, KL divergence...) that evaluate the distance between a bag of words related to a candidate annotation, and the words surrounding the entity to annotate in the text.

In more recent approaches, it is suggested that annotation processes based on similarity distance measures can be improved by making use of other annotations present in the same document. Such techniques are referred to as *semantic relatedness* (Strube and Ponzetto, 2006), *collective disambiguation* (Hoffart et al., 2011b), or *joint disambiguation* (Fahrni et al., 2012). The idea is to evaluate in a set of candidate links which one is the most likely to be correct by taking the other links contained in the document into account. For example, if a NE describes a city name like *Paris*, it is more probable that the correct link for this city name designates *Paris (France)* rather than *Paris (Texas)* if a neighbor entity offers candidate links semantically related to *Paris (France)* like the *Seine river* or the *Champs-Élysées*. Such techniques mostly involve exploration of graphs resulting of all the candidate annotations proposed for a given document, and try to rank the best candidates for each annotation using an ontology. The ontology (like YAGO or DBpedia) provides a pre-

existing set of potential relations between the entities to link (like for instance, in our previous example, *Paris (France) has_river Seine*) that will be used to rank the best candidates according to their mutual presence in the document.

In this paper we explore the capabilities of a disambiguation algorithm using all the available annotation layers of NEs to improve their links. The paper makes the following novel propositions: 1) the ontology used to evaluate the relatedness of candidates is replaced by internal links and categories from the Wikipedia corpus; 2) the coherence of entities is improved prior to the calculation of semantic relatedness using a co-reference resolution algorithm, and a NE label correction method; 3) the proposed method is robust enough to improve the performance of existing entity linking annotation engines, which are capable of providing a set of ranked candidates for each annotation in a document.

This paper is organized as follows. Section 2 describes related works. The proposed method is presented in Section 3 where we explain how our SemLinker system prepares documents that contain mentions to disambiguate, then we detail the disambiguation algorithm. The evaluation of the complete system is provided in Section 4. Finally, we discuss the obtained results, and conclude.

2 Related Work

Entity annotation and linking in natural language text has been extensively studied in NLP research. A strong effort has been conducted recently by the TAC-KBP evaluation task (Ji et al., 2010) to create standardized corpus, and annotation standards based on Wikipedia for evaluation and comparison of EL systems. In this paper, we consider the TAC-KBP framework. We describe below some recent approaches proposed for solving the EL task.

2.1 Wikipedia-based Disambiguation Methods

The use of Wikipedia for explicit disambiguation dates back to (Bunescu and Pasca, 2006) who built a system that compared the context of a mention to the Wikipedia categories of an entity candidate. Lately, (Cucerzan, 2007; Milne and Witten, 2008; Nguyen and Cao, 2008) extended this framework by using richer features for similarity comparison. Some authors like Milne and Witten (2008) utilized machine learning methods rather than a similarity function to map mentions to entities. They

also introduced the notion of semantic relatedness. Alternative propositions were suggested in other works like (Han and Zhao, 2009) that considered the relatedness of common noun phrases in a mention context with Wikipedia article names. While all these approaches focus on semantic relation between entities, their potential is limited by the separate mapping of candidate links for each mention.

2.2 Semantic Web Compliant Methods

More recently, several systems have been launched as web services dedicated to EL tasks. Most of them are compliant with new emergent semantic web standards like LinkedData network. DBpedia Spotlight (Mendes et al., 2011) is a system that finds mentions of DBpedia (Auer et al., 2007) resources in a textual document. Wikimeta (Charton and Gagnon, 2012) is another system relying on DBpedia. It uses bags of words to disambiguate semantic entities according to a cosine similarity algorithm. Those systems have been compared with commercial ones like AlchemyAPI, Zemanta, or Open Calais in (Gangemi, 2013). The study showed that they perform differently on various essential aspects of EL tasks (mention detection, linking, disambiguation). This suggests a wide range of potential improvements on many aspects of the EL task. Only some of these systems introduce the semantic relatedness in their methods like the AIDA (Hoffart et al., 2011b) system. It proposes a disambiguation method that combines popularity-based priors, similarity measures, and coherence. It relies on the Wikipedia-derived YAGO2 (Hoffart et al., 2011a) knowledge base.

3 Proposed Algorithm

We propose a mutual disambiguation algorithm that improves the accuracy of entity links in a document by using successive corrections applied to an *annotation object* representing this document. The annotation object is composed of information extracted from the document along with linguistic and semantic annotations as described hereafter.

3.1 Annotation Object

Documents are processed by an annotator capable of producing POS tags for each word, as well as spans, NE surface forms, NE labels and ranked candidate Wikipedia URIs for each candidate NE. For each document \mathcal{D} , this knowledge is gathered

in an array called *annotation object*, which has initially one row per document lexical unit. Since the system focuses on NEs, rows with lexical units that do not belong to a NE SF are dropped from the annotation object, and NE SF are refined as described in (Charton et al., 2014). When NE SF are spanned over several rows, these rows are merged into a single one. Thus, we consider an annotation object $\mathcal{A}_{\mathcal{D}}$, which is an array with a row for each NE, and columns storing related knowledge.

If n NEs were annotated in \mathcal{D} , then $\mathcal{A}_{\mathcal{D}}$ has n rows. If l candidate URIs are provided for each NE, then $\mathcal{A}_{\mathcal{D}}$ has $(l + 4)$ columns $c_{u,u \in \{1,l+4\}}$. Columns c_1 to c_l store Wikipedia URIs associated with NEs, ordered by decreasing values of likelihood. Column c_{l+1} stores the offset of the NEs, c_{l+2} stores their surface forms, c_{l+3} stores the NE labels (PER, ORG, ...), and c_{l+4} stores the (vectors of) POS tags associated with the NE surface forms. $\mathcal{A}_{\mathcal{D}}$ contains all the available knowledge about the NEs found in \mathcal{D} . Before being processed by the disambiguation module, $\mathcal{A}_{\mathcal{D}}$ is dynamically updated by correction processes.

3.2 Named Entity Label Correction

To support the correction process based on co-reference chains, the system tries to correct NE labels for all the NEs listed in the *annotation object*. The NE label correction process assigns the same NE label to all the NEs associated with the same first rank URI. For all the rows in $\mathcal{A}_{\mathcal{D}}$, sets of rows with identical first rank URIs are considered. Then, for each set, NE labels are counted per type, and all the rows in a same set are updated with the most frequent NE label found in the set, i.e. all the NEs in this set are tagged with this label.

3.3 Correction Based on Co-reference Chains

First rank candidate URIs are corrected by a process that relies on co-reference chains found in the document. The co-reference detection is conducted using the information recorded in the annotation object. Among the NEs present in the document, the ones that co-refer are identified and clustered by logical rules applied to the content of the annotation object. When a co-reference chain of NEs is detected, the system assigns the same URI to all the members of the chain. This URI is selected through a decision process that gives more weight to longer surface forms and frequent URIs. The following example illustrates an application of this correction process:

Three sentences are extracted from a document about Paris, the French capital. NEs are indicated in brackets, first rank URIs and surface forms are added below the content of each sentence.

- [Paris] is famous around the world.

URI₁: http://en.wikipedia.org/wiki/Paris_Hilton

NE surface form: Paris

- The [city of Paris] attracts millions of tourists.

URI₁: <http://en.wikipedia.org/wiki/Paris>

NE surface form: city of Paris

- The [capital of France] is easy to reach by train.

URI₁: <http://en.wikipedia.org/wiki/Paris>

NE surface form: capital of France

The three NEs found in these sentences compose a co-reference chain. The second NE has a longer surface form than the first one, and its associated first rank URI is the most frequent. Hence, the co-reference correction process will assign the right URI to the first NE (URI₁: <http://en.wikipedia.org/wiki/Paris>), which was wrongly linked to the actress Paris Hilton.

3.4 Mutual Disambiguation Process

The extraction of an accurate link is a process occurring after the URI annotation of NEs in the whole document. The system makes use of all the semantic content stored in $\mathcal{A}_{\mathcal{D}}$ to locally improve the precision of each URI annotation in the document. The Mutual Disambiguation Process (MDP) relies on the graph of all the relations (internal links, categories) between Wikipedia content related to the document annotations.

A basic example of semantic relatedness that should be captured is explained hereafter. Let us consider the mention *IBM* in a given document. Candidate NE annotations for this mention can be *International Business Machine* or *International Brotherhood of Magicians*. But if the *IBM* mention co-occurs with a *Thomas Watson, Jr* mention in the document, there will probably be more links between the *International Business Machine* and *Thomas Watson, Jr* related Wikipedia pages than between the *International Brotherhood of Magicians* and *Thomas Watson, Jr* related Wikipedia pages. The purpose of the MDP is to capture this semantic relatedness information contained in the graph of links extracted from Wikipedia pages related to each candidate annotation.

In MDP, for each Wikipedia URI candidate annotation, all the internal links and categories contained in the source Wikipedia document related

to this URI are collected. This information will be used to calculate a weight for each of the l candidate URI annotations of each mention. For a given NE, this weight is expected to measure the mutual relations of a candidate annotation with all the other candidate annotations of NEs in the document. The input of the MDP is an annotation object \mathcal{A}_D with n rows, obtained as explained in Section 3.1. For all $i \in \llbracket 1, n \rrbracket$, $k \in \llbracket 1, l \rrbracket$, we build the set S_i^k , composed of the Wikipedia URIs and categories contained in the source Wikipedia document related to the URI stored in $\mathcal{A}_D[i][k]$ that we will refer to as URI_i^k to ease the reading.

Scoring:

For all $i, j \in \llbracket 1, n \rrbracket$, $k \in \llbracket 1, l \rrbracket$, we want to calculate the weight of mutual relations between the candidate URI_i^k and all the first rank candidates URI_j^1 for $j \neq i$. The calculation combines two scores that we called *direct semantic relation score* (dsr_score) and *common semantic relation score* (csr_score):

- the dsr_score for URI_i^k sums up the number of occurrences of URI_i^k in S_j^1 for all $j \in \llbracket 1, n \rrbracket - \{i\}$.
- the csr_score for URI_i^k sums up the number of common URIs and categories between S_i^k and S_j^1 for all $j \in \llbracket 1, n \rrbracket - \{i\}$.

We assumed the dsr_score was much more semantically significant than the csr_score , and translated this assumption in the weight calculation by introducing two correction parameters α and β used in the final scoring calculation.

Re-ranking:

For all $i \in \llbracket 1, n \rrbracket$, for each set of URIs $\{\text{URI}_i^k, k \in \llbracket 1, l \rrbracket\}$, the re-ranking process is conducted according to the following steps:

For all $i \in I$,

1. $\forall k \in \llbracket 1, l \rrbracket$, calculate $\text{dsr_score}(\text{URI}_i^k)$
2. $\forall k \in \llbracket 1, l \rrbracket$, calculate $\text{csr_score}(\text{URI}_i^k)$
3. $\forall k \in \llbracket 1, l \rrbracket$, calculate $\text{mutual_relation_score}(\text{URI}_i^k) = \alpha \cdot \text{dsr_score}(\text{URI}_i^k) + \beta \cdot \text{csr_score}(\text{URI}_i^k)$
4. re-order $\{\text{URI}_i^k, k \in \llbracket 1, l \rrbracket\}$, by decreasing order of mutual relation score.

In the following, we detail the MDP in the context of a toy example to illustrate how it works. The document contains two sentences, NE mentions are in bold:

IBM has 12 research laboratories worldwide. **Thomas J. Watson, Jr.** became president of the company.

For the first NE mention [**IBM**], \mathcal{A}_D contains two candidate URIs identifying two different resources:

[**IBM**] $\text{URI}_1^1 \equiv$ International Brotherhood of Magicians
 $\text{URI}_2^1 \equiv$ International Business Machines Corporation

For the second NE mention [**Thomas J. Watson, Jr.**], \mathcal{A}_D contains the following candidate URI, which is ranked first:

[**Thomas J. Watson, Jr.**] $\text{URI}_2^1 \equiv$ Thomas Watson, Jr.

S_1^1 gathers URIs and categories contained in the International Brotherhood of Magicians Wikipedia page. S_2^1 is associated to the International Business Machines Corporation, and S_2^1 to the Thomas Watson, Jr. page. $\text{dsr_score}(\text{URI}_1^1)$ sums up the number of occurrences of URI_1^1 in S_j^1 for all $j \in \llbracket 1, n \rrbracket - \{1\}$. Hence, in the current example, $\text{dsr_score}(\text{URI}_1^1)$ is the number of occurrences of URI_1^1 in S_2^1 , namely the number of times the International Brotherhood of Magicians are cited in the Thomas Watson, Jr. page. Similarly, $\text{dsr_score}(\text{URI}_2^1)$ is equal to the number of times the International Business Machines Corporation is cited in the Thomas Watson, Jr. page. $\text{csr_score}(\text{URI}_1^1)$ sums up the number of common URIs and categories between S_1^1 and S_2^1 , i.e. the number of URIs and categories appearing in both International Brotherhood of Magicians and Thomas Watson, Jr. pages. $\text{csr_score}(\text{URI}_2^1)$ counts the number of URIs and categories appearing in both International Business Machines Corporation and Thomas Watson, Jr. pages.

After calculation, we have:

$\text{mutual_relation_score}(\text{URI}_1^1) < \text{mutual_relation_score}(\text{URI}_2^1)$

The candidate URIs for [**IBM**] are re-ranked accordingly, and International Business Machines Corporation becomes its first rank candidate.

4 Experiments and Results

SemLinker has been evaluated on the TAC-KBP 2012 EL task (Charton et al., 2013). In this task, mentions of entities found in a document collection must be linked to entities in a reference KB, or to new named entities discovered in the collection. The document collection built for KBP 2012 contains a combination of newswire articles (News),

SemLinker									TAC-KBP2012 systems				
modules	no disambiguation			MDP only			all modules			1 st	2 nd	3 rd	median
Category	B^{3+P}	B^{3+R}	B^{3+F1}	B^{3+P}	B^{3+R}	B^{3+F1}	B^{3+P}	B^{3+R}	B^{3+F1}	B^{3+F1}	B^{3+F1}	B^{3+F1}	B^{3+F1}
Overall	0.620	0.633	0.626	0.675	0.681	0.678	0.694	0.695	0.695	0.730	0.699	0.689	0.536
PER	0.771	0.791	0.781	0.785	0.795	0.790	0.828	0.838	0.833	0.809	0.840	0.714	0.645
ORG	0.600	0.571	0.585	0.622	0.578	0.599	0.621	0.569	0.594	0.715	0.615	0.717	0.485
GPE	0.412	0.465	0.437	0.570	0.628	0.598	0.574	0.626	0.599	0.627	0.579	0.614	0.428
News	0.663	0.691	0.677	0.728	0.748	0.738	0.750	0.767	0.758	0.782	0.759	0.710	0.574
Web	0.536	0.520	0.528	0.572	0.550	0.561	0.585	0.556	0.570	0.630	0.580	0.508	0.491

Table 1: SemLinker results on the TAC-KBP 2012 test corpus with/out disambiguation modules, and three best results and median from TAC-KBP 2012 systems.

posts to blogs and newsgroups (Web). Given a query that consists of a document with a specified name mention of an entity, the task is to determine the correct node in the reference KB for the entity, adding a new node for the entity if it is not already in the reference KB. Entities can be of type person (PER), organization (ORG), or geopolitical entity (GPE). The reference knowledge base is derived from an October 2008 dump of English Wikipedia, which includes 818,741 nodes. Table 2 provides a breakdown of the queries per categories of entities, and per type of documents.

Category	All	PER	ORG	GPE	News	Web
# queries	2226	918	706	602	1471	755

Table 2: Breakdown of the TAC-KBP 2012 test corpus queries according to entity types, and document categories.

A complete description of these linguistic resources can be found in (Ellis et al., 2011). For the sake of reproducibility, we applied the KBP scoring metric ($B^3 + F$) described in (TAC-KBP, 2012), and we used the KBP scorer¹.

The evaluated system makes use of the Wikimeta annotation engine. The maximum number of candidate URIs is $l = 15$. The MDP correction parameters α and β described in Section 3.4 have been experimentally set to $\alpha = 10$, $\beta = 2$. Table 1 presents the results obtained by the system in three configurations. In the first column, the system is evaluated without the disambiguation module. In the second column, we applied the MDP without correction processes. The system with the complete disambiguation module obtained the results provided in the third column. The three best results and the median from TAC-KBP 2012 systems are shown in the remaining columns for the sake of comparison.

¹<http://www.nist.gov/tac/2013/KBP/EntityLinking/tools.html>

We observe that the complete algorithm (co-references, named entity labels and MDP) provides the best results on PER NE links. On GPE and ORG entities, the simple application of MDP without prior corrections obtains the best results. A slight loss of accuracy is observed on ORG NEs when the MDP is applied with corrections. For those three categories of entities, we show that the complete system improves the performance of a simple algorithm using distance measures. Results on categories News and Web show that the best performance on the whole KBP corpus (without distinction of NE categories) is obtained with the complete algorithm.

5 Conclusion

The presented system provides a robust semantic disambiguation method, based on mutual relation of entities inside a document, using a standard annotation engine. It uses co-reference, NE normalization methods, and Wikipedia internal links as mutual disambiguation resource to improve the annotations. We show that our proposition improves the performance of a standard annotation engine applied to the TAC-KBP evaluation framework. SemLinker is fully implemented, and publicly released as an open source toolkit (<http://code.google.com/p/semlinker>). It has been deployed in the TAC-KBP 2013 evaluation campaign. Our future work will integrate other annotation engines in the system architecture in a collaborative approach.

Acknowledgments

This research was supported as part of Dr Eric Charton’s Mitacs Elevate Grant sponsored by 3CE. Participation of Dr Marie-Jean Meurs was supported by the Genozymes Project funded by Genome Canada & Génome Québec. The Concordia Tsang Lab provided computing resources.

References

- Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.
- Razvan C. Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. ACL.
- Eric Charton and Michel Gagnon. 2012. A disambiguation resource extracted from Wikipedia for semantic annotation. In *Proceedings of LREC 2012*.
- Eric Charton, Marie-Jean Meurs, Ludovic Jean-Louis, and Michel Gagnon. 2013. SemLinker system for KBP2013: A disambiguation algorithm based on mutual relations of semantic annotations inside a document. In *Text Analysis Conference KBP*. U.S. National Institute of Standards and Technology (NIST).
- Eric Charton, Marie-Jean Meurs, Ludovic Jean-Louis, and Michel Gagnon. 2014. Improving Entity Linking using Surface Form Refinement. In *Proceedings of LREC 2014*.
- Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing EMNLP-CoNLL*. ACL.
- Joe Ellis, Xuansong Li, Kira Griffitt, Stephanie M Strassel, and Jonathan Wright. 2011. Linguistic resources for 2012 knowledge base population evaluations. In *Proceedings of TAC-KBP 2012*.
- Angela Fahrni, Thierry Göckel, and Michael Strube. 2012. Hitsmonolingual and cross-lingual entity linking system at tac 2012: A joint approach. In *TAC (Text Analysis Conference) 2012 Workshop*.
- Aldo Gangemi. 2013. A Comparison of Knowledge Extraction Tools for the Semantic Web. In *The 10th Extended Semantic Web Conference (ESWC) 2013*.
- Xianpei Han and Jun Zhao. 2009. Named entity disambiguation by leveraging wikipedia semantic knowledge. In *Proceedings of the Conference on Information and Knowledge Management (CIKM)*. ACM.
- Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, Edwin Lewis-Kelham, Gerard De Melo, and Gerhard Weikum. 2011a. 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. ACM.
- Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenauf, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011b. 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.
- Heng Ji, Ralph Grishman, HT Dang, and K Griffitt. 2010. Overview of the TAC 2010 knowledge base population track. *Proceedings of TAC 2010*.
- Pablo N Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. DBpedia Spotlight: Shedding Light on the Web of Documents. In *The 7th International Conference on Semantic Systems (I-Semantics) 2011*, pages 1–8.
- David N. Milne and Ian H. Witten. 2008. Named entity disambiguation by leveraging wikipedia semantic knowledge. In *Proceedings of the Conference on Information and Knowledge Management (CIKM)*. ACM.
- Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10.
- Hien T. Nguyen and Tru H. Cao. 2008. Named entity disambiguation on an ontology enriched by wikipedia. In *Research, Innovation and Vision for the Future, 2008. RIVF 2008. IEEE International Conference on*, pages 247–254. IEEE.
- Borislav Popov, Atanas Kiryakov, Angel Kirilov, Dimitar Manov, Damyan Ognyanoff, and Miroslav Goranov. 2003. KIM – Semantic annotation platform. *Lecture Notes in Computer Science*, pages 834–849.
- Michael Strube and Simone Paolo Ponzetto. 2006. WikiRelate! Computing Semantic Relatedness Using Wikipedia. In *AAAI*, volume 6, pages 1419–1424.
- TAC-KBP. 2012. Proposed Task Description for Knowledge-Base Population at TAC 2012. In *Proceedings of TAC-KBP 2012*. National Institute of Standards and Technology.

How Well can We Learn Interpretable Entity Types from Text?

Dirk Hovy

Center for Language Technology
University of Copenhagen
Njalsgade 140, 2300 Copenhagen
dirk@cst.dk

Abstract

Many NLP applications rely on type systems to represent higher-level classes. Domain-specific ones are more informative, but have to be manually tailored to each task and domain, making them inflexible and expensive. We investigate a largely unsupervised approach to learning interpretable, domain-specific entity types from unlabeled text. It assumes that any common noun in a domain can function as potential entity type, and uses those nouns as hidden variables in a HMM. To constrain training, it extracts co-occurrence dictionaries of entities and common nouns from the data. We evaluate the learned types by measuring their prediction accuracy for verb arguments in several domains. The results suggest that it is possible to learn domain-specific entity types from unlabeled data. We show significant improvements over an informed baseline, reducing the error rate by 56%.

1 Introduction

Many NLP applications, such as question answering (QA) or information extraction (IE), use type systems to represent relevant semantic classes. Types allow us to find similarities at a higher level to group lexically different entities together. This helps to filter out candidates that violate certain constraints (e.g., in QA, if the intended answer type is PERSON, we can ignore all candidate answers with a different type), but is also used for feature generation and fact-checking.

A central question is: *where do the types come from?* Typically, they come from a hand-constructed set. This has some disadvantages. Domain-general types, such as named entities or WordNet supersenses (Fellbaum, 1998), often fail

to capture critical domain-specific information (in the medical domain, we might want ANTIBIOTIC, SEDATIVE, etc., rather than just ARTIFACT). Domain-specific types perform much better (Ferrucci et al., 2010), but must be manually adapted to each new domain, which is expensive. Alternatively, unsupervised approaches (Ritter et al., 2010) can be used to learn clusters of similar words, but the resulting types (=cluster numbers) are not human-interpretable, which makes analysis difficult. Furthermore, it requires us to define the number of clusters beforehand.

Ideally, we would like to learn domain-specific types directly from data. To this end, pattern-based approaches have long been used to induce type systems (Hearst, 1992; Kozareva et al., 2008). Recently, Hovy et al. (2011) proposed an approach that uses co-occurrence patterns to find entity type candidates, and then learns their applicability to relation arguments by using them as latent variables in a first-order HMM. However, they only evaluate their method using human sensibility judgements for one domain. While this shows that the types are coherent, it does not tell us much about their applicability.

We extend their approach with three important changes:

1. we evaluate the types by measuring accuracy when using them in an extrinsic task,
2. we evaluate on more than one domain, and
3. we explore a variety of different models.

We measure prediction accuracy when using the learned types in a selectional restriction task for frequent verbs. E.g., given the relation *throw(X, pass)* in the football domain, we compare the model prediction to the gold data $X=QUARTERBACK$. The results indicate that the learned types can be used to in relation extraction tasks.

Our contributions in this paper are:

- we empirically evaluate an approach to learning types from unlabeled data
- we investigate several domains and models
- the learned entity types can be used to predict selectional restrictions with high accuracy

2 Related Work

In relation extraction, we have to identify the relation elements, and then map the arguments to types. We follow an open IE approach (Banko and Etzioni, 2008) and use dependencies to identify the elements. In contrast to most previous work (Pardo et al., 2006; Yao et al., 2011; Yao et al., 2012), we have **no** pre-defined set of types, but try to learn it along with the relations. Some approaches use types from general data bases such as Wikipedia, Freebase, etc. (Yan et al., 2009; Eichler et al., 2008; Syed and Viegas, 2010), side-stepping the question how to construct those DBs in the first place. We are less concerned with extraction performance, but focus on the accuracy of the learned type system by measuring how well it performs in a prediction task.

Talukdar et al. (2008) and Talukdar and Pereira (2010) present graph-based approaches to the similar problem of class-instance learning. While this provides a way to discover types, it requires a large graph that does not easily generalize to new instances (transductive), since it produces no predictive model. The models we use are transductive and can be applied to unseen data. Our approach follows Hovy et al. (2011). However, they only evaluate one model on football by collecting sensibility ratings from Mechanical Turk. Our method provides extrinsic measures of performance on several domains.

3 Model

Our goal is to find semantic type candidates in the data, and apply them in relation extraction to see which ones are best suited. We restrict ourselves to verbal relations. We build on the approach by Hovy et al. (2011), which we describe briefly below. It consists of two parts: extracting the type candidates and fitting the model.

The basic idea is that semantic types are usually common nouns, often frequent ones from the

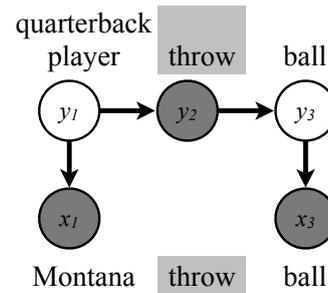


Figure 1: Example of input sentence x and output types for the HMM. Note that the verb type is treated as observed variable.

domain at hand. Thus all common nouns are possible types, and can be used as latent variables in an HMM. By estimating emission and transition parameters with EM, we can learn the subset of nouns to apply.

However, assuming the set of all common nouns as types is intractable, and would not allow for efficient learning. To restrict the search space and improve learning, we first have to learn which types modify entities and record their co-occurrence, and use this as dictionary.

Kleiman: professor:25, expert:13, (*specialist:1*)

Tilton: executive:37, economist:17, (*chairman:4, president:2*)

Figure 2: Examples of dictionary entries with counts. Types in brackets are not considered.

Dictionary Construction The number of common nouns in a domain is generally too high to consider all of them for every entity. A common way to restrict the number of types is to provide a dictionary that lists all legal types for each entity (Merialdo, 1994; Ravi and Knight, 2009; Täckström et al., 2013). To construct this dictionary, we collect for each entity (i.e., a sequence of words labeled with NNP or NNPS tags) in our data all common nouns (NN, NNS) that modify it. These are

1. nominal modifiers (“*judge Scalosi ...*”),
2. appositions (“*Tilton, a professor at ...*”), and
3. copula constructions (“*Finton, who is the investor ...*”).

These modifications can be collected from the dependency parse trees. For each entity, we store the

type candidates and their associated counts. See Figure 2 for examples. We only consider types observed more than 10 times. Any entity without type information, as well as dictionary entities with only singleton types are treated as unknown tokens (“UNK”). We map UNK to the 50 most common types in the dictionary. Verbs are considered to each have their own type, i.e., token and label for verbs are the same. We do not modify this step.

Original Model Hovy et al. (2011) construct a HMM using subject-verb-object (SVO) parse triples as observations, and the type candidates as hidden variables. Similar models have been used in (Abney and Light, 1999; Pardo et al., 2006). We estimate the free model parameters with EM (Dempster et al., 1977), run for a fixed number of iterations (30) or until convergence.

Note that Forward-backward EM has time complexity of $\mathcal{O}(N^2T)$, where N is the number of states, and T the number of time steps. $T = 3$ in the model formulations used here, but N is much larger than typically found in NLP tasks (see also Table 3). The only way to make this tractable is to restrict the free parameters the model needs to estimate to the transitions.

The model is initialized by jointly normalizing¹ the dictionary counts to obtain the emission parameters, which are then fixed (except for the unknown entities ($P(\text{word} = \text{UNK} | \text{type} = \cdot)$). Transition parameters are initialized uniformly (restricted to potentially observable type sequences), and kept as free parameters for the model to optimize.

Common nouns can be both hidden variables and observations in the model, so they act like annotated items: their legal types are restricted to the identity. All entities are thus constrained by the dictionary, as in (Merialdo, 1994). To further constrain the model, only the top three types of each entity are considered. Since the type distribution typically follows a Zipf curve, this still captures most of the information.

¹This preserves the observed entity-specific distributions. Under conditional normalization, the type candidates from frequent entities tend to dominate those of infrequent entities. I.e., the model favors an unlikely candidate for entity a if it is frequent for entity b .

The model can be fully specified as

$$P(\mathbf{x}, \mathbf{y}) = P(y_1) \cdot P(x_1 | y_1) \prod_{i=2}^3 P(y_i | y_{i-1}) \cdot P(x_i | y_i) \quad (1)$$

where \mathbf{x} is an input triple of a verb and its arguments, and \mathbf{y} a sequence of types.

4 Extending the Model

The model used by Hovy et al. (2011) was a simple first order HMM, with the elements in SVO order (see Figure 3a). We observe two points: we always deal with the same number of elements, and we have observed variables. We can thus move from a sequential model to a general graphical model by adding transitions and re-arranging the structure.

Since we do not model verbs (they each have their identity as type), they act like observed variables. We can thus move them in first position and condition the subject on it (3b).

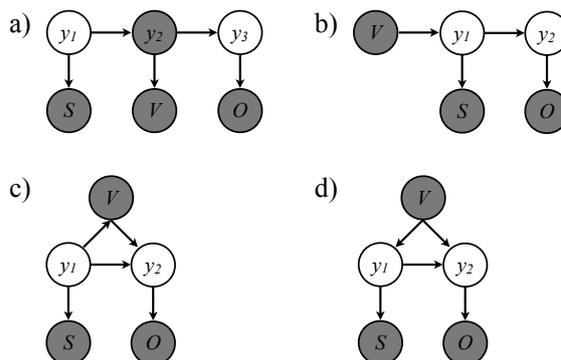


Figure 3: Original SVO. model (a), modified VSO order (b), extension to general models (c and d)

By adding additional transitions, we can constrain the latent variables further. This is similar to moving from a first to a second order HMM. In contrast to the original model, we also distinguish between unknown entities in the first and second argument position.

The goal of these modifications is to restrict the number of potential values for the argument positions. This allows us to use the models to type individual instances. In contrast, the objective in Hovy et al. (2011) was to collect frequent relation templates from a domain to populate a knowledge base.

The modifications presented here extend to

system	Football				Finances				Law			
	arg1	arg2	avg	ΔBL	arg1	arg2	avg	ΔBL	arg1	arg2	avg	ΔBL
baseline	0.28	0.26	0.27	—	0.39	0.42	0.41	—	0.37	0.32	0.35	—
orig.	0.05	0.23	0.14	-0.13	0.08	0.39	0.23	-0.18	0.06	0.31	0.18	-0.17
VSO, seq.	<i>0.37</i>	0.28	0.32	+0.05	0.38	0.45	0.41	0.0	<i>0.45</i>	0.37	<i>0.41</i>	+0.06
SVO, net	<i>0.63</i>	0.60	0.62	+0.35	<i>0.55</i>	0.63	0.59	+0.18	<i>0.69</i>	0.68	0.68	+0.33
VSO, net	0.66	<i>0.58</i>	0.62	+0.35	0.61	<i>0.54</i>	<i>0.57</i>	+0.16	0.71	<i>0.62</i>	<i>0.66</i>	+0.31

Table 1: Accuracy for most frequent sense baseline and different models on three domains. Italic numbers denote significant improvement over baseline (two-tailed t-test at $p < 0.01$). ΔBL = difference to baseline.

system	Football			Finances			Law		
	arg1	arg2	avg	arg1	arg2	avg	arg1	arg2	avg
orig.	0.17	0.38	0.27	0.18	0.52	0.35	0.17	0.48	0.32
VSO, seq.	0.56	0.42	0.49	0.55	0.58	0.57	0.61	0.51	0.56
SVO, net	0.75	0.69	0.72	0.68	0.73	0.71	0.78	0.77	0.78
VSO, net	0.78	0.67	0.72	0.74	0.66	0.70	0.81	0.72	0.76

Table 2: Mean reciprocal rank for models on three domains.

verbs with more than two arguments, but in the present paper, we focus on binary relations.

5 Experiments

Since the labels are induced dynamically from the data, traditional precision/recall measures, which require a known ground truth, are difficult to obtain. Hovy et al. (2011) measured sensibility by obtaining human ratings and measuring weighted accuracies over all relations. While this gives an intuition of the general methodology, it is harder to put in context. Here, we want to evaluate the model’s performance in a downstream task. We measure its ability to predict the correct types for verbal arguments. We evaluate on three different domains.

As test case, we use a cloze test, or fill-in-the-blank. We select instances that contain a type-candidate word in subject or object position and replace that word with the unknown token. We can then compare the model’s prediction to the original word to measure accuracy.

5.1 Data

Like Yao et al. (2012) and Hovy et al. (2011), we derive our data from the New York Times (NYT) corpus (Sandhaus, 2008). It contains several years worth of articles, manually annotated with meta-data such as author, content, etc. Similar to Yao et al. (2012), we use articles whose *content* meta-

data field contains certain labels to distinguish data from different domains. We use the labels *Football*², *Law and Legislation*, and *Finances*.

We remove meta-data and lists, tokenize, parse, and lemmatize all articles. We then automatically extract subject-verb-object (SVO) triples from the parses, provided the verb is a full verb. Similarly to (Pardo et al., 2006), we focus on the top 100 full verbs for efficiency reasons, though nothing in our approach prevents us from extending it to all verbs. For each domain, we select all instances which have a potential type (common noun) in at least one argument position. These serve as corpus.

	Football	Finances	Law
unique types	7,139	18,186	10,618
unique entities	38,282	27,528	12,782

Table 3: Statistics for the three domains.

As test data, we randomly select a subset of 1000 instances for each argument, provided they contain one of the 50 most frequent types in subject or object position, such as *player* in “*player* throw pass”. This serves as gold data. We then replace those types by UNK (i.e., we get “UNK throw pass”) and use this as test set for our model.³

Table 3 shows that the domains vary with re-

²The data likely differs from Hovy et al. (2011).

³We omit cases with two unknown arguments, since this

spect to the ratio of unique types to unique entities. Football uses many different entities (e.g., team and player names), but has few types (e.g., player positions), while the other domains use more types, but fewer entities (e.g., company names, law firms, etc.).

5.2 Evaluation

We run Viterbi decoding on each test set with our trained model to predict the most likely type for the unknown entities. We then compare these predictions to the type in the respective gold data and compute the accuracy for each argument position. As baseline, we predict the argument types most frequently observed for the particular verb in training, e.g., predict *PLAYER* as subject of *tackle* in football. We evaluate the influence of the different model structures on performance.

6 Results

Table 1 shows the accuracy of the different models in the prediction task for the three different domains. The low results of the informed baseline indicate the task complexity.

We note that the original model, a bigram HMM with SVO order (Figure 3a), fails to improve accuracy over the baseline (although its overall results were judged sensible). Changing the input order to VSO (Figure 3b) improves accuracy for both arguments over SVO order and the baseline, albeit not significantly. The first argument gains more, since conditioning the subject type on the (unambiguous) verb is more constrained than starting out with the subject. Conditioning the object directly upon the subject creates sparser bigrams, which capture “who does what to whom”.

Moving from the HMMs to a general graphical model structure (Figures 3c and d) creates a sparser distribution and significantly improves accuracy across the board. Again, the position of the verb makes a difference: in SVO order, accuracy for the second argument is better, while in VSO order, accuracy for the subject increases. This indicates that direct conditioning on the verb is the strongest predictor. Intuitively, knowing the verb restricts the possible arguments much more than knowing the arguments restrict the possible verbs (the types of entities who can throw something are

becomes almost impossible to predict without further context, even for humans (compare “UNK make UNK”).

limited, but knowing that the subject is a quarterback still allows all kinds of actions).

We also compute the mean reciprocal rank (MRR) for each condition (see Table 2). MRR denotes the inverse rank in the model’s k -best output at which the correct answer occurs, i.e., $\frac{1}{k}$. The result gives us an intuition of “how far off” the model predictions are. Across domains, the correct answer is found on average among the top two (rank 1.36). Note that since MRR require k -best outputs, we cannot compute a measure for the baseline.

7 Conclusion

We evaluated an approach to learning domain-specific interpretable entity types from unlabeled data. Type candidates are collected from patterns and modeled as hidden variables in graphical models. Rather than using human sensibility judgments, we evaluate prediction accuracy for selectional restrictions when using the learned types in three domains. The best model improves 35 percentage points over an informed baseline. On average, we reduce the error rate by 56%. We conclude that it is possible to learn interpretable type systems directly from data.

Acknowledgements

The author would like to thank Victoria Fossum, Eduard Hovy, Kevin Knight, and the anonymous reviewers for their invaluable feedback.

References

- Steven Abney and Marc Light. 1999. Hiding a semantic hierarchy in a Markov model. In *Proceedings of the ACL Workshop on Unsupervised Learning in Natural Language Processing*, volume 67.
- Michele Banko and Oren. Etzioni. 2008. The trade-offs between open and traditional relation extraction. *Proceedings of ACL-08: HLT*, pages 28–36.
- Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.
- Kathrin Eichler, Holmer Hemsén, and Günter Neumann. 2008. Unsupervised relation extraction from web documents. *LREC*. <http://www.lrecconf.org/proceedings/lrec2008>.
- Christiane Fellbaum. 1998. *WordNet: an electronic lexical database*. MIT Press USA.

- David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. 2010. Building Watson: An overview of the DeepQA project. *AI magazine*, 31(3):59–79.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics.
- Dirk Hovy, Chunliang Zhang, Eduard Hovy, and Anselmo Peñas. 2011. Unsupervised discovery of domain-specific knowledge from text. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1466–1475, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Zornitsa Kozareva, Ellen Riloff, and Eduard Hovy. 2008. Semantic class learning from the web with hyponym pattern linkage graphs. *Proceedings of ACL-08: HLT*, pages 1048–1056.
- Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational linguistics*, 20(2):155–171.
- Thiago Pardo, Daniel Marcu, and Maria Nunes. 2006. Unsupervised Learning of Verb Argument Structures. *Computational Linguistics and Intelligent Text Processing*, pages 59–70.
- Sujith Ravi and Kevin Knight. 2009. Minimized Models for Unsupervised Part-of-Speech Tagging. 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 504–512. Association for Computational Linguistics.
- Alan Ritter, Mausam, and Oren Etzioni. 2010. A latent dirichlet allocation method for selectional preferences. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 424–434, Uppsala, Sweden, July. Association for Computational Linguistics.
- Evan Sandhaus, editor. 2008. *The New York Times Annotated Corpus*. Number LDC2008T19. Linguistic Data Consortium, Philadelphia.
- Zareen Syed and Evelyne Viegas. 2010. A hybrid approach to unsupervised relation discovery based on linguistic analysis and semantic typing. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading*, pages 105–113. Association for Computational Linguistics.
- Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. *Transactions of the ACL*.
- Partha P. Talukdar and Fernando Pereira. 2010. Experiments in graph-based semi-supervised learning methods for class-instance acquisition. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1473–1481. Association for Computational Linguistics.
- Partha P. Talukdar, Joseph Reisinger, Marcus Pasca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. Weakly-supervised acquisition of labeled class instances using graph random walks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 582–590. Association for Computational Linguistics.
- Yulan Yan, Naoaki Okazaki, Yutaka Matsuo, Zhenglu Yang, and Mitsuru Ishizuka. 2009. Unsupervised relation extraction by mining wikipedia texts using information from the web. 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: Volume 2-Volume 2*, pages 1021–1029. Association for Computational Linguistics.
- Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured relation discovery using generative models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1456–1466. Association for Computational Linguistics.
- Limin Yao, Sebastian Riedel, and Andrew McCallum. 2012. Unsupervised relation discovery with sense disambiguation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 712–720. Association for Computational Linguistics.

Learning Translational and Knowledge-based Similarities from Relevance Rankings for Cross-Language Retrieval

Shigehiko Schamoni and Felix Hieber and Artem Sokolov and Stefan Riezler

Department of Computational Linguistics

Heidelberg University, 69120 Heidelberg, Germany

{schamoni, hieber, sokolov, riezler}@cl.uni-heidelberg.de

Abstract

We present an approach to cross-language retrieval that combines dense knowledge-based features and sparse word translations. Both feature types are learned directly from relevance rankings of bilingual documents in a pairwise ranking framework. In large-scale experiments for patent prior art search and cross-lingual retrieval in Wikipedia, our approach yields considerable improvements over learning-to-rank with either only dense or only sparse features, and over very competitive baselines that combine state-of-the-art machine translation and retrieval.

1 Introduction

Cross-Language Information Retrieval (CLIR) for the domain of web search successfully leverages state-of-the-art Statistical Machine Translation (SMT) to either produce a single most probable translation, or a weighted list of alternatives, that is used as search query to a standard search engine (Chin et al., 2008; Ture et al., 2012). This approach is advantageous if large amounts of in-domain sentence-parallel data are available to train SMT systems, but relevance rankings to train retrieval models are not.

The situation is different for CLIR in special domains such as patents or Wikipedia. Parallel data for translation have to be extracted with some effort from comparable or noisy parallel data (Utiyama and Isahara, 2007; Smith et al., 2010), however, relevance judgments are often straightforwardly encoded in special domains. For example, in patent prior art search, patents granted at any patent office worldwide are considered relevant if they constitute prior art with respect to the invention claimed in the query patent. Since patent applicants and lawyers are required to list

relevant prior work explicitly in the patent application, patent citations can be used to automatically extract large amounts of relevance judgments across languages (Graf and Azzopardi, 2008). In Wikipedia search, one can imagine a Wikipedia author trying to investigate whether a Wikipedia article covering the subject the author intends to write about already exists in another language. Since authors are encouraged to avoid orphan articles and to cite their sources, Wikipedia has a rich linking structure between related articles, which can be exploited to create relevance links between articles across languages (Bai et al., 2010).

Besides a rich citation structure, patent documents and Wikipedia articles contain a number of further cues on relatedness that can be exploited as features in learning-to-rank approaches. For monolingual patent retrieval, Guo and Gomes (2009) and Oh et al. (2013) advocate the use of dense features encoding domain knowledge on inventors, assignees, location and date, together with dense similarity scores based on bag-of-word representations of patents. Bai et al. (2010) show that for the domain of Wikipedia, learning a sparse matrix of word associations between the query and document vocabularies from relevance rankings is useful in monolingual and cross-lingual retrieval. Sokolov et al. (2013) apply the idea of learning a sparse matrix of bilingual phrase associations from relevance rankings to cross-lingual retrieval in the patent domain. Both show improvements of learning-to-rank on relevance data over SMT-based approaches on their respective domains.

The main contribution of this paper is a thorough evaluation of dense and sparse features for learning-to-rank that have so far been used only monolingually or only on either patents or Wikipedia. We show that for both domains, patents and Wikipedia, jointly learning bilingual sparse word associations and dense knowledge-based similarities directly on relevance ranked

data improves significantly over approaches that use either only sparse or only dense features, and over approaches that combine query translation by SMT with standard retrieval in the target language. Furthermore, we show that our approach can be seen as supervised model combination that allows to combine SMT-based and ranking-based approaches for further substantial improvements. We conjecture that the gains are due to orthogonal information contributed by domain-knowledge, ranking-based word associations, and translation-based information.

2 Related Work

CLIR addresses the problem of translating or projecting a query into the language of the document repository across which retrieval is performed. In a *direct translation* approach (DT), a state-of-the-art SMT system is used to produce a single best translation that is used as search query in the target language. For example, Google’s CLIR approach combines their state-of-the-art SMT system with their proprietary search engine (Chin et al., 2008).

Alternative approaches avoid to solve the hard problem of word reordering, and instead rely on token-to-token translations that are used to project the query terms into the target language with a probabilistic weighting of the standard term tf-idf scheme. Darwish and Oard (2003) termed this method the *probabilistic structured query* approach (PSQ). The advantage of this technique is an implicit query expansion effect due to the use of probability distributions over term translations (Xu et al., 2001). Ture et al. (2012) brought SMT back into this paradigm by projecting terms from n -best translations from synchronous context-free grammars.

Ranking approaches have been presented by Guo and Gomes (2009) and Oh et al. (2013). Their method is a classical learning-to-rank setup where pairwise ranking is applied to a few hundred dense features. Methods to learn sparse word-based translation correspondences from supervised ranking signals have been presented by Bai et al. (2010) and Sokolov et al. (2013). Both approaches work in a cross-lingual setting, the former on Wikipedia data, the latter on patents.

Our approach extends the work of Sokolov et al. (2013) by presenting an alternative learning-to-rank approach that can be used for supervised model combination to integrate dense and sparse

features, and by evaluating both approaches on cross-lingual retrieval for patents and Wikipedia. This relates our work to supervised model merging approaches (Sheldon et al., 2011).

3 Translation and Ranking for CLIR

SMT-based Models. We will refer to DT and PSQ as SMT-based models that translate a query, and then perform monolingual retrieval using BM25. Translation is agnostic of the retrieval task.

Linear Ranking for Word-Based Models. Let $\mathbf{q} \in \{0, 1\}^Q$ be a query and $\mathbf{d} \in \{0, 1\}^D$ be a document where the j^{th} vector dimension indicates the occurrence of the j^{th} word for dictionaries of size Q and D . A linear ranking model is defined as

$$f(\mathbf{q}, \mathbf{d}) = \mathbf{q}^\top W \mathbf{d} = \sum_{i=1}^Q \sum_{j=1}^D q_i W_{ij} d_j,$$

where $W \in \mathbb{R}^{Q \times D}$ encodes a matrix of ranking-specific word associations (Bai et al., 2010). We optimize this model by pairwise ranking, which assumes labeled data in the form of a set \mathcal{R} of tuples $(\mathbf{q}, \mathbf{d}^+, \mathbf{d}^-)$, where \mathbf{d}^+ is a relevant (or higher ranked) document and \mathbf{d}^- an irrelevant (or lower ranked) document for query \mathbf{q} . The goal is to find a weight matrix W such that an inequality $f(\mathbf{q}, \mathbf{d}^+) > f(\mathbf{q}, \mathbf{d}^-)$ is violated for the fewest number of tuples from \mathcal{R} . We present two methods for optimizing W in the following.

Pairwise Ranking using Boosting (BM). The Boosting-based Ranking baseline (Freund et al., 2003) optimizes an exponential loss:

$$\mathcal{L}_{exp} = \sum_{(\mathbf{q}, \mathbf{d}^+, \mathbf{d}^-) \in \mathcal{R}} \mathcal{D}(\mathbf{q}, \mathbf{d}^+, \mathbf{d}^-) e^{f(\mathbf{q}, \mathbf{d}^-) - f(\mathbf{q}, \mathbf{d}^+)},$$

where $\mathcal{D}(\mathbf{q}, \mathbf{d}^+, \mathbf{d}^-)$ is a non-negative importance function on tuples. The algorithm of Sokolov et al. (2013) combines batch boosting with bagging over a number of independently drawn bootstrap data samples from \mathcal{R} . In each step, the single word pair feature is selected that provides the largest decrease of \mathcal{L}_{exp} . The found corresponding models are averaged. To reduce memory requirements we used random feature hashing with the size of the hash of 30 bits (Shi et al., 2009). For regularization we rely on early stopping.

Pairwise Ranking with SGD (VW). The second objective is an ℓ_1 -regularized hinge loss:

$$\mathcal{L}_{hng} = \sum_{(\mathbf{q}, \mathbf{d}^+, \mathbf{d}^-) \in \mathcal{R}} (f(\mathbf{q}, \mathbf{d}^+) - f(\mathbf{q}, \mathbf{d}^-))_+ + \lambda \|W\|_1,$$

where $(x)_+ = \max(0, 1 - x)$ and λ is the regularization parameter. This newly added model utilizes the standard implementation of online SGD from the Vowpal Wabbit (VW) toolkit (Goel et al., 2008) and was run on a data sample of 5M to 10M tuples from \mathcal{R} . On each step, W is updated with a scaled gradient vector $\nabla_W \mathcal{L}_{hng}$ and clipped to account for ℓ_1 -regularization. Memory usage was reduced using the same hashing technique as for boosting.

Domain Knowledge Models. Domain knowledge features for patents were inspired by Guo and Gomes (2009): a feature fires if two patents share similar aspects, e.g. a common inventor. As we do not have access to address data, we omit geolocation features and instead add features that evaluate similarity w.r.t. patent classes extracted from IPC codes. Documents within a patent section, i.e. the topmost hierarchy, are too diverse to provide useful information but more detailed classes and the count of matching classes do.

For Wikipedia, we implemented features that compare the relative length of documents, number of links and images, the number of common links and common images, and Wikipedia categories: Given the categories associated with a foreign query, we use the language links on the Wikipedia category pages to generate a set of “translated” English categories S . The English-side category graph is used to construct sets of super- and subcategories related to the candidate document’s categories. This expansion is done in both directions for two levels resulting in 5 category sets. The intersection between target set T_n and the source category set S reflects the category level similarity between query and document, which we calculate as a mutual containment score $s_n = \frac{1}{2}(|S \cap T_n|/|S| + |S \cap T_n|/|T_n|)$ for $n \in \{-2, -1, 0, +1, +2\}$ (Broder, 1997).

Optimization for these additional models including domain knowledge features was done by overloading the vector representation of queries \mathbf{q} and documents \mathbf{d} in the VW linear learner: Instead of sparse word-based features, \mathbf{q} and \mathbf{d} are represented by real-valued vectors of dense domain-knowledge features. Optimization for the overloaded vectors is done as described above for VW.

4 Model Combination

Combination by Borda Counts. The baseline consensus-based voting Borda Count procedure

endows each voter with a fixed amount of voting points which he is free to distribute among the scored documents (Aslam and Montague, 2001; Sokolov et al., 2013). The aggregate score for two rankings $f_1(\mathbf{q}, \mathbf{d})$ and $f_2(\mathbf{q}, \mathbf{d})$ for all (\mathbf{q}, \mathbf{d}) in the test set is then a simple linear interpolation: $f_{agg}(\mathbf{q}, \mathbf{d}) = \kappa \frac{f_1(\mathbf{q}, \mathbf{d})}{\sum_{\mathbf{d}} f_1(\mathbf{q}, \mathbf{d})} + (1 - \kappa) \frac{f_2(\mathbf{q}, \mathbf{d})}{\sum_{\mathbf{d}} f_2(\mathbf{q}, \mathbf{d})}$. Parameter κ was adjusted on the dev set.

Combination by Linear Learning. In order to acquire the best combination of more than two models, we created vectors of model scores along with domain knowledge features and reused the VW pairwise ranking approach. This means that the vector representation of queries \mathbf{q} and documents \mathbf{d} in the VW linear learner is overloaded once more: In addition to dense domain-knowledge features, we incorporate arbitrary ranking models as dense features whose value is the score of the ranking model. Training data was sampled from the dev set and processed with VW.

5 Data

Patent Prior Art Search (JP-EN). We use BoostCLIR¹, a Japanese-English (JP-EN) corpus of patent abstracts from the MAREC and NTCIR data (Sokolov et al., 2013). It contains automatically induced relevance judgments for patent abstracts (Graf and Azzopardi, 2008): EN patents are regarded as relevant with level (3) to a JP query patent, if they are in a family relationship (e.g., same invention), cited by the patent examiner (2), or cited by the applicant (1). Statistics on the ranking data are given in Table 1. On average, queries and documents contain about 5 sentences.

Wikipedia Article Retrieval (DE-EN). The intuition behind our Wikipedia retrieval setup is as follows: Consider the situation where the German (DE) Wikipedia article on geological sea *stacks* does not yet exist. A native speaker of German with profound knowledge in geology intends to write it, naming it “*Brandungspfeiler*”, while seeking to align its structure with the EN counterpart. The task of a CLIR engine is to return relevant EN Wikipedia articles that may describe the very same concept (*Stack (geology)*), or relevant instances of it (*Bako National Park, Lange Anna*). The information need may be paraphrased as a high-level definition of the topic. Since typically the first sentence of any Wikipedia article is such

¹www.cl.uni-heidelberg.de/boostclir

	#q	#d	#d ⁺ /q	#words/q
Patents (JP-EN)				
train	107,061	888,127	13.28	178.74
dev	2,000	100,000	13.24	181.70
test	2,000	100,000	12.59	182.39
Wikipedia (DE-EN)				
train	225,294	1,226,741	13.04	25.80
dev	10,000	113,553	12.97	25.75
test	10,000	115,131	13.22	25.73

Table 1: Ranking data statistics: number of queries and documents, avg. number of relevant documents per query, avg. number of words per query.

a well-formed definition, this allows us to extract a large set of one sentence queries from Wikipedia articles. For example: “*Brandungspfeiler sind vor einer Kliffküste aufragende Felsentürme und vergleichbare Formationen, die durch Brandungserosion gebildet werden.*”² Similar to Bai et al. (2010) we induce relevance judgments by aligning DE queries with their EN counterparts (“mates”) via the graph of inter-language links available in articles and Wikidata³. We assign relevance level (3) to the EN mate and level (2) to all other EN articles that link to the mate, *and* are linked by the mate. Instead of using all outgoing links from the mate, we only use articles with bidirectional links.

To create this data⁴ we downloaded XML and SQL dumps of the DE and EN Wikipedia from, resp., 22nd and 4th of November 2013. Wikipedia markup removal and link extraction was carried out using the Cloud9 toolkit⁵. Sentence extraction was done with NLTK⁶. Since Wikipedia articles vary greatly in length, we restricted EN documents to the first 200 words after extracting the link graph to reduce the number of features for BM and VW models. To avoid rendering the task too easy for literal keyword matching of queries about named entities, we removed title words from the German queries. Statistics are given in Table 1.

Preprocessing Ranking Data. In addition to lowercasing and punctuation removal, we applied Correlated Feature Hashing (CFH), that makes collisions more likely for words with close meaning (Bai et al., 2010). For patents, vocabularies contained 60k and 365k words for JP and EN. Filtering special symbols and stopwords reduced the JP vocabulary size to 50k (small enough not to resort to CFH). To reduce the EN vocabulary

²de.wikipedia.org/wiki/Brandungspfeiler

³www.wikidata.org/

⁴www.cl.uni-heidelberg.de/wiki/clir

⁵lintool.github.io/Cloud9/index.html

⁶www.nltk.org/

to a comparable size, we applied similar preprocessing *and* CFH with $F=30k$ and $k=5$. Since for Wikipedia data, the DE and EN vocabularies were both large (6.7M and 6M), we used the same filtering and preprocessing as for the patent data before applying CFH with $F=40k$ and $k=5$ on both sides.

Parallel Data for SMT-based CLIR. For both tasks, DT and PSQ require an SMT baseline system trained on parallel corpora that are disjoint from the ranking data. A JP-EN system was trained on data described and preprocessed by Sokolov et al. (2013), consisting of 1.8M parallel sentences from the NTCIR-7 JP-EN PatentMT subtask (Fujii et al., 2008) and 2k parallel sentences for parameter development from the NTCIR-8 test collection. For Wikipedia, we trained a DE-EN system on 4.1M parallel sentences from Europarl, Common Crawl, and News-Commentary. Parameter tuning was done on 3k parallel sentences from the WMT’11 test set.

6 Experiments

Experiment Settings. The SMT-based models use `cdec` (Dyer et al., 2010). Word alignments were created with `mgiza` (JP-EN) and `fast_align` (Dyer et al., 2013) (DE-EN). Language models were trained with the KenLM toolkit (Heafield, 2011). The JP-EN system uses a 5-gram language model from the EN side of the training data. For the DE-EN system, a 4-gram model was built on the EN side of the training data and the EN Wikipedia documents. Weights for the standard feature set were optimized using `cdec`’s MERT (JP-EN) and MIRA (DE-EN) implementations (Och, 2003; Chiang et al., 2008). PSQ on patents reuses settings found by Sokolov et al. (2013); settings for Wikipedia were adjusted on its dev set ($n=1000$, $\lambda=0.4$, $L=0$, $C=1$).

Patent retrieval for DT was done by sentence-wise translation and subsequent re-joining to form one query per patent, which was ranked against the documents using BM25. For PSQ, BM25 is computed on expected term and document frequencies.

For ranking-based retrieval, we compare several combinations of learners and features (Table 2). VW denotes a sparse model using word-based features trained with SGD. BM denotes a similar model trained using Boosting. DK denotes VW training of a model that represents queries q and documents d by dense domain-knowledge features instead of by sparse word-based vectors. In

order to simulate pass-through behavior of out-of-vocabulary terms in SMT systems, additional features accounting for source and target term identity were added to DK and BM models. The parameter λ for VW was found on dev set. Statistical significance testing was performed using the paired randomization test (Smucker et al., 2007).

Borda denotes model combination by Borda Count voting where the linear interpolation parameter is adjusted for MAP on the respective development sets with grid search. This type of model combination only allows to combine pairs of rankings. We present a combination of SMT-based CLIR, DT+PSQ, a combination of dense and sparse features, DK+VW, and a combination of both combinations, (DT+PSQ)+(DK+VW).

LinLearn denotes model combination by overloading the vector representation of queries \mathbf{q} and documents \mathbf{d} in the VW linear learner by incorporating arbitrary ranking models as dense features. In difference to grid search for *Borda*, optimal weights for the linear combination of incorporated ranking models can be learned automatically. We investigate the same combinations of ranking models as described for *Borda* above. We do not report combination results including the sparse BM model since they were consistently lower than the ones with the sparse VW model.

Test Results. Experimental results on test data are given in Table 2. Results are reported with respect to MAP (Manning et al., 2008), NDCG (Järvelin and Kekäläinen, 2002), and PRES (Magdy and Jones, 2010). Scores were computed on the top 1,000 retrieved documents.

As can be seen from inspecting the two blocks of results, one for patents, one for Wikipedia, we find the same system rankings on both datasets. In both cases, as *standalone* systems, DT and PSQ are very close and far better than any ranking approach, irrespective of the objective function or the choice of sparse or dense features. Model combination of similar models, e.g., DT and PSQ, gives minimal gains, compared to combining orthogonal models, e.g. DK and VW. The best result is achieved by combining DT and PSQ with DK and VW. This is due to the already high scores of the combined models, but also to the combination of yet other types of orthogonal information. *Borda* voting gives the best result under MAP which is probably due to the adjustment of the interpolation parameter for MAP on the development set.

		combination	models	MAP	NDCG	PRES	
Patents (JP-EN)	<i>standalone</i>		DT	0.2554	0.5397	0.5680	
			PSQ	0.2659	0.5508	0.5851	
			DK	0.2203	0.4874	0.5171	
			VW	0.2205	0.4989	0.4911	
			BM	0.1669	0.4167	0.4665	
	<i>Borda</i>		DT+PSQ	*0.2747	*0.5618	*0.5988	
			DK+VW	*0.3023	*0.5980	*0.6137	
			(DT+PSQ)+(DK+VW)	*0.3465	*0.6420	*0.6858	
		<i>LinLearn</i>		DT+PSQ	†*0.2707	†*0.5578	†*0.5941
				DK+VW	†*0.3283	†*0.6366	†*0.7104
	DT+PSQ+DK+VW		†* 0.3739	†* 0.6755	†* 0.7599		
Wikipedia (DE-EN)	<i>standalone</i>		DT	0.3678	0.5691	0.7219	
			PSQ	0.3642	0.5671	0.7165	
			DK	0.2661	0.4584	0.6717	
			VW	0.1249	0.3389	0.6466	
			BM	0.1386	0.3418	0.6145	
	<i>Borda</i>		DT+PSQ	*0.3742	*0.5777	*0.7306	
			DK+VW	*0.3238	*0.5484	*0.7736	
			(DT+PSQ)+(DK+VW)	* 0.4173	*0.6333	*0.8031	
		<i>LinLearn</i>		DT+PSQ	†*0.3718	†*0.5751	†*0.7251
				DK+VW	†*0.3436	†*0.5686	†*0.7914
	DT+PSQ+DK+VW		*0.4137	†* 0.6435	†* 0.8233		

Table 2: Test results for *standalone* CLIR models using direct translation (DT), probabilistic structured queries (PSQ), sparse model with CFH (VW), sparse boosting model (BM), dense domain knowledge features (DK), and model combinations using Borda Count voting (*Borda*) or linear supervised model combination (*LinLearn*). Significant differences (at $p=0.01$) between aggregated systems and all its components are indicated by *, between *LinLearn* and the respective *Borda* system by †.

Under NDCG and PRES, *LinLearn* achieves the best results, showing the advantage of automatically learning combination weights that leads to stable results across various metrics.

7 Conclusion

Special domains such as patents or Wikipedia offer the possibility to extract cross-lingual relevance data from citation and link graphs. These data can be used to directly optimizing cross-lingual ranking models. We showed on two different large-scale ranking scenarios that a supervised combination of orthogonal information sources such as domain-knowledge, translation knowledge, and ranking-specific word associations by far outperforms a pipeline of query translation and retrieval. We conjecture that if these types of information sources are available, a supervised ranking approach will yield superior results in other retrieval scenarios as well.

Acknowledgments

This research was supported in part by DFG grant RI-2221/1-1 “Cross-language Learning-to-Rank for Patent Retrieval”.

References

- Javed A. Aslam and Mark Montague. 2001. Models for metasearch. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, New Orleans, LA.
- Bing Bai, Jason Weston, David Grangier, Ronan Collobert, Kunihiko Sadamasa, Yanjun Qi, Olivier Chapelle, and Kilian Weinberger. 2010. Learning to rank with (a lot of) word features. *Information Retrieval Journal*, 13(3):291–314.
- Andrei Z. Broder. 1997. On the resemblance and containment of documents. In *Compression and Complexity of Sequences (SEQUENCES'97)*, pages 21–29. IEEE Computer Society.
- David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'08)*, Waikiki, Hawaii.
- Jeffrey Chin, Maureen Heymans, Alexandre Kojoukhov, Jocelyn Lin, and Hui Tan. 2008. Cross-language information retrieval. Patent Application. US 2008/0288474 A1.
- Kareem Darwish and Douglas W. Oard. 2003. Probabilistic structured query methods. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'03)*, Toronto, Canada.
- Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the ACL 2010 System Demonstrations*, Uppsala, Sweden.
- Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM Model 2. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, GA.
- Yoav Freund, Ray Iyer, Robert E. Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969.
- Atsushi Fujii, Masao Utiyama, Mikio Yamamoto, and Takehito Utsuro. 2008. Overview of the patent translation task at the NTCIR-7 workshop. In *Proceedings of NTCIR-7 Workshop Meeting*, Tokyo, Japan.
- Sharad Goel, John Langford, and Alexander L. Strehl. 2008. Predictive indexing for fast search. In *Advances in Neural Information Processing Systems*, Vancouver, Canada.
- Erik Graf and Leif Azzopardi. 2008. A methodology for building a patent test collection for prior art search. In *Proceedings of the 2nd International Workshop on Evaluating Information Access (EVIA'08)*, Tokyo, Japan.
- Yunsong Guo and Carla Gomes. 2009. Ranking structured documents: A large margin based approach for patent prior art search. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'09)*, Pasadena, CA.
- Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation (WMT'11)*, Edinburgh, UK.
- Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions in Information Systems*, 20(4):422–446.
- Walid Magdy and Gareth J.F. Jones. 2010. PRES: a score metric for evaluating recall-oriented information retrieval applications. In *Proceedings of the ACM SIGIR conference on Research and development in information retrieval (SIGIR'10)*, New York, NY.
- Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Meeting on Association for Computational Linguistics (ACL'03)*, Sapporo, Japan.
- Sooyoung Oh, Zhen Lei, Wang-Chien Lee, Prasenjit Mitra, and John Yen. 2013. CV-PCR: A context-guided value-driven framework for patent citation recommendation. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'13)*, San Francisco, CA.
- Daniel Sheldon, Milad Shokouhi, Martin Szummer, and Nick Craswell. 2011. Lambdamerge: Merging the results of query reformulations. In *Proceedings of WSDM'11*, Hong Kong, China.
- Qinfeng Shi, James Petterson, Gideon Dror, John Langford, Alexander J. Smola, Alexander L. Strehl, and Vishy Vishwanathan. 2009. Hash Kernels. In *Proceedings of the 12th Int. Conference on Artificial Intelligence and Statistics (AISTATS'09)*, Irvine, CA.
- Jason R. Smith, Chris Quirk, and Kristina Toutanova. 2010. Extracting parallel sentences from comparable corpora using document level alignment. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'10)*, Los Angeles, CA.

- Mark D. Smucker, James Allan, and Ben Carterette. 2007. A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of the 16th ACM conference on Conference on Information and Knowledge Management (CIKM '07)*, New York, NY.
- Artem Sokolov, Laura Jehl, Felix Hieber, and Stefan Riezler. 2013. Boosting cross-language retrieval by learning bilingual phrase associations from relevance rankings. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'13)*.
- Ferhan Ture, Jimmy Lin, and Douglas W. Oard. 2012. Combining statistical translation techniques for cross-language information retrieval. In *Proceedings of the International Conference on Computational Linguistics (COLING'12)*, Bombay, India.
- Masao Utiyama and Hitoshi Isahara. 2007. A Japanese-English patent parallel corpus. In *Proceedings of MT Summit XI*, Copenhagen, Denmark.
- Jinxi Xu, Ralph Weischedel, and Chanh Nguyen. 2001. Evaluating a probabilistic model for cross-lingual information retrieval. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, New York, NY.

Two-Stage Hashing for Fast Document Retrieval

Hao Li* Wei Liu[†] Heng Ji*

*Computer Science Department,

Rensselaer Polytechnic Institute, Troy, NY, USA

{lih13, jih}@rpi.edu

[†]IBM T. J. Watson Research Center, Yorktown Heights, NY, USA

weiliu@us.ibm.com

Abstract

This work fulfills sublinear time Nearest Neighbor Search (NNS) in massive-scale document collections. The primary contribution is to propose a two-stage unsupervised hashing framework which harmoniously integrates two state-of-the-art hashing algorithms Locality Sensitive Hashing (LSH) and Iterative Quantization (ITQ). LSH accounts for neighbor candidate pruning, while ITQ provides an efficient and effective reranking over the neighbor pool captured by LSH. Furthermore, the proposed hashing framework capitalizes on both term and topic similarity among documents, leading to precise document retrieval. The experimental results convincingly show that our hashing based document retrieval approach well approximates the conventional Information Retrieval (IR) method in terms of retrieving semantically similar documents, and meanwhile achieves a speedup of over one order of magnitude in query time.

1 Introduction

A *Nearest Neighbor Search* (NNS) task aims at searching for top K objects (*e.g.*, documents) which are most similar, based on pre-defined similarity metrics, to a given query object in an existing dataset. NNS is essential in dealing with many search related tasks, and also fundamental to a broad range of Natural Language Processing (NLP) down-stream problems including person name spelling correction (Udupa and Kumar, 2010), document translation pair acquisition (Krstovski and Smith, 2011), large-scale similar noun list generation (Ravichandran et al., 2005), lexical variants mining (Gouws et al., 2011), and large-scale first story detection (Petrovic et al., 2010).

Hashing has recently emerged to be a popular solution to tackling fast NNS, and been successfully applied to a variety of non-NLP problems such as visual object detection (Dean et al., 2013) and recognition (Torralba et al., 2008a; Torralba et al., 2008b), large-scale image retrieval (Kulis and Grauman, 2012; Liu et al., 2012; Gong et al., 2013), and large-scale machine learning (Weiss et al., 2008; Liu et al., 2011; Liu, 2012). However, hashing has received limited attention in the NLP field to the date. The basic idea of hashing is to represent each data object as a binary code (each bit of a code is one digit of “0” or “1”). When applying hashing to handle NLP problems, the advantages are two-fold: 1) the capability to store a large quantity of documents in the main memory. for example, one can store 250 million documents with 1.9G memory using only 64 bits for each document while a large news corpus such as the English Gigaword fifth edition¹ stores 10 million documents in a 26G hard drive; 2) the time efficiency of manipulating binary codes, for example, computing the hamming distance between a pair of binary codes is several orders of magnitude faster than computing the real-valued cosine similarity over a pair of document vectors.

The early explorations of hashing focused on using random permutations or projections to construct randomized hash functions, *e.g.*, the well-known Min-wise Hashing (MinHash) (Broder et al., 1998) and Locality Sensitive Hashing (LSH) (Andoni and Indyk, 2008). In contrast to such data-independent hashing schemes, recent research has been geared to studying data-dependent hashing through learning compact hash codes from a training dataset. The state-of-the-art unsupervised learning-based hashing methods include Spectral Hashing (SH) (Weiss et al., 2008), Anchor Graph Hashing (AGH) (Liu et al., 2011), and Iterative Quantization (ITQ) (Gong et al.,

¹<http://catalog.ldc.upenn.edu/LDC2011T07>

2013), all of which endeavor to make the learned hash codes preserve or reveal some intrinsic structure, such as local neighborhood structure, low-dimensional manifolds, or the closest hypercube, underlying the training data. Despite achieving data-dependent hash codes, most of these “learning to hash” methods cannot guarantee a high success rate of looking a query code up in a hash table (referred to as hash table lookup in literature), which is critical to the high efficacy of exploiting hashing in practical uses. It is worth noting that we choose to use ITQ in the proposed two-stage hashing framework for its simplicity and efficiency. ITQ has been found to work better than SH by Gong et al. (2013), and be more efficient than AGH in terms of training time by Liu (2012).

To this end, in this paper we propose a novel two-stage unsupervised hashing framework to simultaneously enhance the hash lookup success rate and increase the search accuracy by integrating the advantages of both LSH and ITQ. Furthermore, we make the hashing framework applicable to combine different similarity measures in NNS.

2 Background and Terminology

- **Binary Codes:** A bit (a single bit is “0” or “1”) sequence assigned to represent a data object. For example, represent a document as a 8-bit code “11101010”.
- **Hash Table:** A linear table in which all binary codes of a data set are arranged to be table indexes, and each table bucket contains the IDs of the data items sharing the same code.
- **Hamming Distance:** The number of bit positions in which bits of the two codes differ.
- **Hash Table Lookup:** Given a query q with its binary code h_q , find the candidate neighbors in a hash table such that the Hamming distances from their codes to h_q are no more than a small distance threshold ϵ . In practice ϵ is usually set to 2 to maintain the efficiency of table lookups.
- **Hash Table Lookup Success Rate:** Given a query q with its binary code h_q , the probability to find at least one neighbor in the table buckets whose corresponding codes (*i.e.*, indexes) are within a Hamming ball of radius ϵ centered at h_q .
- **Hamming Ranking:** Given a query q with its binary code h_q , rank all data items according to the Hamming distances between their

codes and h_q ; the smaller the Hamming distance, the higher the data item is ranked.

3 Document Retrieval with Hashing

In this section, we first provide an overview of applying hashing techniques to a document retrieval task, and then introduce two unsupervised hashing algorithms: LSH acts as a neighbor-candidate filter, while ITQ works towards precise reranking over the candidate pool returned by LSH.

3.1 Document Retrieval

The most traditional way of retrieving nearest neighbors for documents is to represent each document as a term vector of which each element is the *tf-idf* weight of a term. Given a query document vector q , we use the *Cosine* similarity measure to evaluate the similarity between q and a document x in a dataset:

$$\text{sim}(q, x) = \frac{q^\top x}{\|q\| \|x\|}. \quad (1)$$

Then the traditional document retrieval method exhaustively scans all documents in the dataset and returns the most similar ones. However, such a brute-force search does not scale to massive datasets since the search time complexity for each query is $O(n)$; additionally, the computational cost spent on Cosine similarity calculation is also nontrivial.

3.2 Locality Sensitive Hashing

The core idea of LSH is that if two data points are close, then after a “projection” operation they will remain close. In other words, similar data points are more likely to be mapped into the same bucket with a high collision probability. In a typical LSH setting of k bits and L hash tables, a query point $q \in \mathbb{R}^d$ and a dataset point $x \in \mathbb{R}^d$ collide if and only if

$$h_{ij}(q) \equiv h_{ij}(x), i \in [1 : L], j \in [1 : k], \quad (2)$$

where the hash function $h_{ij}(\cdot)$ is defined as

$$h_{ij}(x) = \text{sgn}(w_{ij}^\top x), \quad (3)$$

in which $w_{ij} \in \mathbb{R}^d$ is a random projection direction with components being independently and identically drawn from a normal distribution, and the sign function $\text{sgn}(x)$ returns 1 if $x > 0$ and -1 otherwise. Note that we use “1/-1” bits for derivations and training, and “1/0” bits for the hashing

implementation including converting data to binary codes, arranging binary codes into hash tables, and hash table lookups.

3.3 Iterative Quantization

The central idea of ITQ is to learn the binary codes achieving the lowest quantization error that encoding raw data to binary codes incurs. This is pursued by seeking a rotation of the zero-centered projected data. Suppose that a set of n data points $\mathcal{X} = \{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^n$ are provided. The data matrix is defined as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$. In order to reduce the data dimension from d to the desired code length $c < d$, Principal Component Analysis (PCA) or Latent Semantic Analysis (LSA) is first applied to \mathbf{X} . We thus obtain the zero-centered projected data matrix as $\mathbf{V} = (\mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\top)\mathbf{X}\mathbf{U}$ where $\mathbf{U} \in \mathbb{R}^{d \times c}$ is the projection matrix.

After the projection operation, ITQ minimizes the quantization error as follows

$$\mathbb{Q}(\mathbf{B}, \mathbf{R}) = \|\mathbf{B} - \mathbf{V}\mathbf{R}\|_{\text{F}}^2, \quad (4)$$

where $\mathbf{B} \in \{1, -1\}^{n \times c}$ is the code matrix each row of which contains a binary code, $\mathbf{R} \in \mathbb{R}^{c \times c}$ is the target orthogonal rotation matrix, and $\|\cdot\|_{\text{F}}$ denotes the Frobenius norm. Finding a local minimum of the quantization error in Eq. (4) begins with a random initialization of \mathbf{R} , and then employs a K-means clustering like iterative procedure. In each iteration, each (projected) data point is assigned to the nearest vertex of the binary hypercube, and \mathbf{R} always satisfying $\mathbf{R}\mathbf{R}^\top = \mathbf{I}$ is subsequently updated to minimize the quantization loss given the current assignment; the two steps run alternately until a convergence is encountered. Concretely, the two updating steps are:

1. **Fix \mathbf{R} and update \mathbf{B} :** minimize the following quantization loss

$$\begin{aligned} \mathbb{Q}(\mathbf{B}, \mathbf{R}) &= \|\mathbf{B}\|_{\text{F}}^2 + \|\mathbf{V}\mathbf{R}\|_{\text{F}}^2 - 2\text{tr}(\mathbf{R}^\top \mathbf{V}^\top \mathbf{B}) \\ &= nc + \|\mathbf{V}\|_{\text{F}}^2 - 2\text{tr}(\mathbf{R}^\top \mathbf{V}^\top \mathbf{B}) \\ &= \text{constant} - 2\text{tr}(\mathbf{R}^\top \mathbf{V}^\top \mathbf{B}), \end{aligned} \quad (5)$$

achieving $\mathbf{B} = \text{sgn}(\mathbf{V}\mathbf{R})$;

2. **Fix \mathbf{B} and update \mathbf{R} :** perform the SVD of the matrix $\mathbf{V}^\top \mathbf{B} \in \mathbb{R}^{c \times c}$ to obtain $\mathbf{V}^\top \mathbf{B} = \mathbf{S}\mathbf{\Omega}\mathbf{\hat{S}}^\top$, and then set $\mathbf{R} = \mathbf{\hat{S}}\mathbf{\Omega}^\top$.

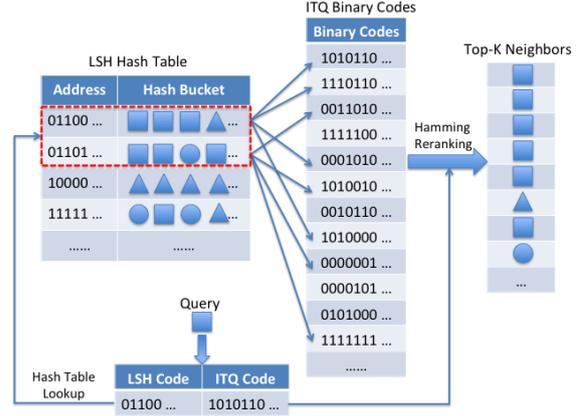


Figure 1: The two-stage hashing framework.

3.4 Two-Stage Hashing

There are three main merits of LSH. (1) It tries to preserve the Cosine similarity of the original data with a probabilistic guarantee (Charikar, 2002). (2) It is training free, and thus very efficient in hashing massive databases to binary codes. (3) It has a very high hash table lookup success rate. For example, in our experiments LSH with more than one hash table is able to achieve a perfect 100% hash lookup success rate. Unfortunately, its drawback is the low search precision that is observed even with long hash bits and multiple hash tables.

ITQ tries to minimize the quantization error of encoding data to binary codes, so its advantage is the high quality (potentially high precision of Hamming ranking) of the produced binary codes. Nevertheless, ITQ frequently suffers from a poor hash lookup success rate when longer bits (*e.g.*, ≥ 48) are used (Liu, 2012). For example, in our experiments ITQ using 384 bits has a 18.47% hash lookup success rate within Hamming radius 2. Hence, Hamming ranking (costing $O(n)$ time) must be invoked for the queries for which ITQ fails to return any neighbors via hash table lookup, which makes the searches inefficient especially on very large datasets.

Taking into account the above advantages and disadvantages of LSH and ITQ, we propose a two-stage hashing framework to harmoniously integrate them. Fig. 1 illustrates our two-stage framework with a toy example where identical shapes denote ground-truth nearest neighbors.

In this framework, LSH accounts for neighbor candidate pruning, while ITQ provides an efficient and effective reranking over the neighbor pool captured by LSH. To be specific, the pro-

posed framework enjoys two advantages:

1. Provide a simple solution to accomplish both a high hash lookup success rate and high precision, which does not require scanning the whole list of the ITQ binary codes but scanning the short list returned by LSH hash table lookup. Therefore, a high hash lookup success rate is attained by the LSH stage, while maintaining high search precision due to the ITQ reranking stage.

2. Enable a hybrid hashing scheme combining two similarity measures. The term similarity is used during the LSH stage that directly works on document tf-idf vectors; during the ITQ stage, the topic similarity is used since ITQ works on the topic vectors obtained by applying Latent semantic analysis (LSA) (Deerwester et al., 1990) to those document vectors. LSA (or PCA), the first step in running ITQ, can be easily accelerated via a simple sub-selective sampling strategy which has been proven theoretically and empirically sound by Li et al. (2014). As a result, the nearest neighbors returned by the two-stage hashing framework turns out to be both lexically and topically similar to the query document. To summarize, the proposed two-stage hashing framework works in an unsupervised manner, achieves a sublinear search time complexity due to LSH, and attains high search precision thanks to ITQ. After hashing all data (documents) to LSH and ITQ binary codes, we do not need to save the raw data in memory. Thus, our approach can scale to gigantic datasets with compact storage and fast search speed.

4 Experiments

Data and Evaluations

For the experiments, we use the English portion of the standard TDT-5 dataset, which consists of 278,109 documents from a time spanning April 2003 to September 2003. 126 topics are annotated with an average of 51 documents per topic, and other unlabeled documents are irrelevant to them. We select six largest topics for the top-K NNS evaluation, with each including more than 250 documents. We randomly select 60 documents from each of the six topics for testing. The six topics are (1). Bombing in Riyadh, Saudi Arabia (2). Mad cow disease in North America (3). Casablanca bombs (4). Swedish Foreign Minister killed (5). Liberian former president arrives in exile and (6). UN official killed in attack. For each

document, we apply the Stanford Tokenizer² for tokenization; remove stopwords based on the stop list from InQuery (Callan et al., 1992), and apply Porter Stemmer (Porter, 1980) for stemming.

If one retrieved document shares the same topic label with the query document, they are true neighbors. We evaluate the precision of the top-K candidate documents returned by each method and calculate the average precision across all queries.

Results

We first evaluate the quality of term vectors and ITQ binary codes by conducting the whole list Cosine similarity ranking and hamming distance ranking, respectively. For each query document, the top-K candidate documents with highest Cosine similarity scores and shortest hamming distances are returned, then we calculate the average precision for each K. Fig. 2(a) demonstrates that ITQ binary codes could preserve document similarities as traditional term vectors. It is interesting to notice that ITQ binary codes are able to outperform traditional term vectors. It is mainly because some documents are topically related but share few terms thus their relatedness can be captured by LSA. Fig. 2(a) also shows that the NNS precision keep increasing as longer ITQ code length is used and is converged when ITQ code length equals to 384 bits. Therefore we set ITQ code length as 384 bits in the rest of the experiments.

Fig. 2(b) - Fig. 2(e) show that our two-stage hashing framework surpasses LSH with a large margin for both small K (*e.g.*, $K \leq 10$) and large K (*e.g.*, $K \geq 100$) in top-K NNS. It also demonstrates that our hashing based document retrieval approach with only binary codes from LSH and ITQ well approximates the conventional IR method. Another crucial observation is that with ITQ reranking, a small number of LSH hash tables is needed in the pruning step. For example, LSH(40bits) + ITQ(384bits) and LSH(48bits) + ITQ(384bits) are able to reach convergence with only four LSH hash tables. In that case, we can alleviate one main drawback of LSH as it usually requires a large number of hash tables to maintain the hashing quality.

Since the LSH pruning time can be ignored, the search time of the two-stage hashing scheme equals to the time of hamming distance reranking in ITQ codes for all candidates produced from LSH pruning step, *e.g.*, LSH(48bits, 4 tables) +

²<http://nlp.stanford.edu/software/corenlp.shtml>

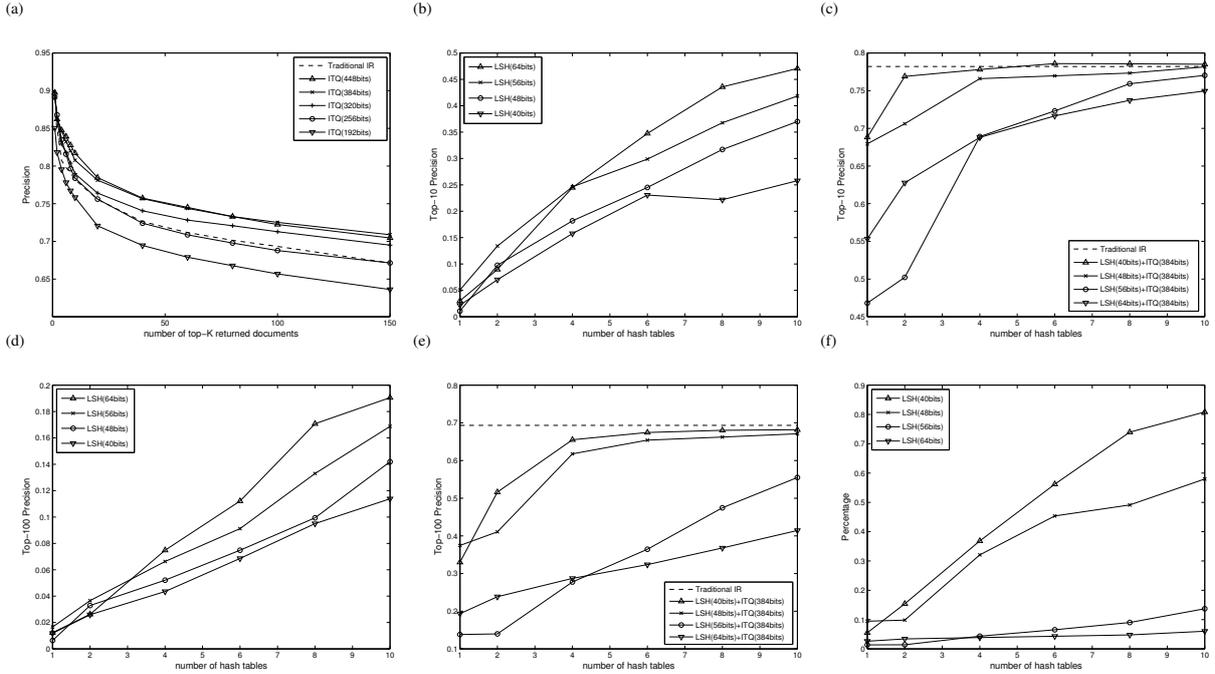


Figure 2: (a) ITQ code quality for different code length, (b) LSH Top-10 Precision, (c) LSH + ITQ(384bits) Top-10 Precision, (d) LSH Top-100 Precision, (e) LSH + ITQ(384bits) Top-100 Precision, (f) The percentage of visited data samples by LSH hash lookup.

ITQ(384bits) takes only one thirtieth of the search time as the traditional IR method. Fig. 2 (f) shows the ITQ data reranking percentage for different LSH bit lengths and table numbers. As the LSH bit length increases or the hash table number decreases, a lower percentage of the candidates will be selected for reranking, and thus costs less search time.

The percentage of visited data samples by LSH hash lookup is a key factor that influence the NNS precision in the two-stage hashing framework. Generally, higher rerank percentage results in better top-K NNS Precision. Further more, by comparing Fig. 2 (c) and (e), it shows that our framework works better for small K than for large K. For example, scanning 5.52% of the data is enough for achieving similar top-10 NNS result as the traditional IR method while 36.86% of the data is needed for top-100 NNS. The reason of the lower performance with large K is that some true neighbors with the same topic label do not share high term similarities and may be filtered out in the LSH step when the rerank percentage is low.

5 Conclusion

In this paper, we proposed a novel two-stage unsupervised hashing framework for efficient and effective nearest neighbor search in massive docu-

ment collections. The experimental results have shown that this framework achieves not only comparable search accuracy with the traditional IR method in retrieving semantically similar documents, but also an order of magnitude speedup in search time. Moreover, our approach can combine two similarity measures in a hybrid hashing scheme, which is beneficial to comprehensively modeling the document similarity. In our future work, we plan to design better data representation which can well fit into the two-stage hashing theme; we also intend to apply the proposed hashing approach to more informal genres (*e.g.*, tweets) and other down-stream NLP applications (*e.g.*, first story detection).

Acknowledgements

This work was supported by the U.S. ARL No. W911NF-09-2-0053 (NSCTA), NSF IIS-0953149, DARPA No. FA8750-13-2-0041, IBM, Google and RPI. 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 U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

- A. Andoni and P. Indyk. 2008. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, 51(1):117–122.
- A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. 1998. Min-wise independent permutations. In *Proc. STOC*.
- J. P. Callan, W. B. Croft, and S. M. Harding. 1992. The inquiry retrieval system. In *Proc. the Third International Conference on Database and Expert Systems Applications*.
- M. Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proc. STOC*.
- T. Dean, M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. 2013. Fast, accurate detection of 100,000 object classes on a single machine. In *Proc. CVPR*.
- S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. 1990. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407.
- Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. 2013. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2916–2929.
- S. Gouws, D. Hovy, and D. Metzler. 2011. Unsupervised mining of lexical variants from noisy text. In *Proc. EMNLP*.
- K. Krstovski and D. A. Smith. 2011. A minimally supervised approach for detecting and ranking document translation pairs. In *Proc. the sixth ACL Workshop on Statistical Machine Translation*.
- B. Kulis and K. Grauman. 2012. Kernelized locality-sensitive hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1092–1104.
- Y. Li, C. Chen, W. Liu, and J. Huang. 2014. Sub-selective quantization for large-scale image search. In *Proc. AAAI Conference on Artificial Intelligence (AAAI)*.
- W. Liu, J. Wang, S. Kumar, and S.-F. Chang. 2011. Hashing with graphs. In *Proc. ICML*.
- W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang. 2012. Supervised hashing with kernels. In *Proc. CVPR*.
- W. Liu. 2012. Large-scale machine learning for classification and search. In *PhD Thesis, Graduate School of Arts and Sciences, Columbia University*.
- S. Petrovic, M. Osborne, and V. Lavrenko. 2010. Streaming first story detection with application to twitter. In *Proc. HLT-NAACL*.
- M. F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.
- D. Ravichandran, P. Pantel, and E. H. Hovy. 2005. Randomized algorithms and nlp: Using locality sensitive hash functions for high speed noun clustering. In *Proc. ACL*.
- A. Torralba, R. Fergus, and W. T. Freeman. 2008a. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970.
- A. Torralba, R. Fergus, and Y. Weiss. 2008b. Small codes and large image databases for recognition. In *Proc. CVPR*.
- R. Udupa and S. Kumar. 2010. Hashing-based approaches to spelling correction of personal names. In *Proc. EMNLP*.
- Y. Weiss, A. Torralba, and R. Fergus. 2008. Spectral hashing. In *NIPS 21*.

An Annotation Framework for Dense Event Ordering

Taylor Cassidy

IBM Research

taylor.cassidy.ctr@mail.mil

Bill McDowell

Carnegie Mellon University

forkunited@gmail.com

Nathanael Chambers

US Naval Academy

nchamber@usna.edu

Steven Bethard

Univ. of Alabama at Birmingham

bethard@cis.uab.edu

Abstract

Today's event ordering research is heavily dependent on annotated corpora. Current corpora influence shared evaluations and drive algorithm development. Partly due to this dependence, most research focuses on *partial orderings* of a document's events. For instance, the TempEval competitions and the TimeBank only annotate small portions of the event graph, focusing on the most salient events or on specific types of event pairs (e.g., only events in the same sentence). Deeper temporal reasoners struggle with this sparsity because the entire temporal picture is not represented. This paper proposes a new annotation process with a mechanism to force annotators to label connected graphs. It generates 10 times more relations per document than the TimeBank, and our *TimeBank-Dense* corpus is larger than all current corpora. We hope this process and its dense corpus encourages research on new global models with deeper reasoning.

1 Introduction

The TimeBank Corpus (Pustejovsky et al., 2003) ushered in a wave of data-driven event ordering research. It provided for a common dataset of relations between events and time expressions that allowed the community to compare approaches. Later corpora and competitions have based their tasks on the TimeBank setup. This paper addresses one of its shortcomings: sparse annotation. We describe a new annotation framework (and a *TimeBank-Dense* corpus) that we believe is needed to fulfill the data needs of deeper reasoners.

The TimeBank includes a small subset of all possible relations in its documents. The annotators were instructed to label relations critical to the document's understanding. The result is a sparse labeling that leaves much of the document unlabeled. The TempEval contests have largely followed suit and focused on specific types of event pairs. For instance, TempEval (Verhagen et al., 2007) only labeled relations between events that syntactically dominated each other. This paper is the first attempt to annotate a document's entire temporal graph.

A consequence of focusing on all relations is a shift from the traditional *classification* task, where the system is given a pair of events and asked only to label the type of relation, to an *identification* task, where the system must determine for itself which events in the document to pair up. For example, in TempEval-1 and 2 (Verhagen et al., 2007; Verhagen et al., 2010), systems were given event pairs in specific syntactic positions: events and times in the same noun phrase, main events in consecutive sentences, etc. We now aim for a shift in the community wherein all pairs are considered candidates for temporal ordering, allowing researchers to ask questions such as: how must algorithms adapt to label the complete graph of pairs, and if the more difficult and ambiguous event pairs are included, how must feature-based learners change?

We are not the first to propose these questions, but this paper is the first to directly propose the means by which they can be addressed. The stated goal of TempEval-3 (UzZaman et al., 2013) was to focus on relation identification instead of classification, but the training and evaluation data followed the TimeBank approach where only a subset of event pairs were labeled. As a result, many systems focused on classification, with the top system classifying pairs in only three syntactic constructions

Current Systems & Evaluations

There were four or five people inside, and they just **started firing**

Ms. Sanders was **hit** several times and was **pronounced dead** at the scene.

The other customers **fled**, and the police **said** it did not **appear** that anyone else was **injured**.

This Proposal

There were four or five people inside, and they just **started firing**

Ms. Sanders was **hit** several times and was **pronounced dead** at the scene.

The other customers **fled**, and the police **said** it did not **appear** that anyone else was **injured**.

Figure 1: A TimeBank annotated document is on the left, and this paper’s TimeBank-Dense annotation is on the right. Solid arrows indicate BEFORE relations and dotted arrows indicate INCLUDED_IN relations.

(Bethard, 2013). We describe the first annotation framework that forces annotators to annotate all pairs¹. With this new process, we created a dense ordering of document events that can properly evaluate both relation identification and relation annotation. Figure 1 illustrates one document before and after our new annotations.

2 Previous Annotation Work

The majority of corpora and competitions for event ordering contain sparse annotations. Annotators for the original TimeBank (Pustejovsky et al., 2003) only annotated relations judged to be salient by the annotator. Subsequent TempEval competitions (Verhagen et al., 2007; Verhagen et al., 2010; Uz-Zaman et al., 2013) mostly relied on the TimeBank, but also aimed to improve coverage by annotating relations between all events and times *in the same sentence*. However, event tokens that were mentioned fewer than 20 times were excluded and only one TempEval task considered relations between events in different sentences. In practical terms, the resulting evaluations remained sparse.

A major dilemma underlying these sparse tasks is that the unlabeled event/time pairs are ambiguous. Each unlabeled pair holds 3 possibilities:

1. The annotator looked at the pair of events and decided that no temporal relation exists.
2. The annotator did not look at the pair of events, so a relation may or may not exist.
3. The annotator failed to look at the pair of events, so a single relation may exist.

Training and evaluation of temporal reasoners is hampered by this ambiguity. To combat this, our

	Events	Times	ReIs	R
TimeBank	7935	1414	6418	0.7
Bramsen 2006	627	–	615	1.0
TempEval-07	6832	1249	5790	0.7
TempEval-10	5688	2117	4907	0.6
TempEval-13	11145	2078	11098	0.8
Kolomiyets-12	1233	–	1139	0.9
Do 2012 ²	324	232	3132	5.6
This work	1729	289	12715	6.3

Table 1: Events, times, relations and the ratio of relations to events + times (R) in various corpora.

annotation adopts the VAGUE relation introduced by TempEval 2007, and our approach forces annotators to use it. This is the only work that includes such a mechanism.

This paper is not the first to look into more dense annotations. Bramsen et al. (2006) annotated multi-sentence segments of text to build directed acyclic graphs. Kolomiyets et al. (2012) annotated “temporal dependency structures”, though they only focused on relations between pairs of events. Do et al. (2012) produced the densest annotation, but “the annotator was not required to annotate all pairs of event mentions, but as many as possible”. The current paper takes a different tack to annotation by *requiring* annotators to label every possible pair of events/times in a given window. Thus this work is the first annotation effort that can guarantee its event/time graph to be strongly connected.

Table 1 compares the size and density of our corpus to others. Ours is the densest and it contains the largest number of temporal relations.

¹As discussed below, all pairs in a given window size.

²Do et al. (2012) reports 6264 relations, but this includes both the relations and their inverses. We thus halve the count

3 A Framework for Dense Annotation

Frameworks for annotating text typically have two independent facets: (1) the practical means of how to label the text, and (2) the higher-level rules about when something should be labeled. The first is often accomplished through a markup language, and we follow prior work in adopting TimeML here. The second facet is the focus of this paper: *when* should an annotator label an ordering relation?

Our proposal starts with documents that have already been annotated with events, time expressions, and document creation times (DCT). The following sentence serves as our motivating example:

*Police **confirmed Friday** that the body **found** along a highway in San Juan **belonged** to Jorge Hernandez.*

This sentence is represented by a 4 node graph (3 events and 1 time). In a completely annotated graph it would have 6 edges (relations) connecting the nodes. In the TimeBank, from which this sentence is drawn, only 3 of the 6 edges are labeled.

The impact of these annotation decisions (i.e., when to annotate a relation) can be significant. In this example, a learner must somehow deal with the 3 unlabeled edges. One option is to assume that they are vague or ambiguous. However, all 6 edges have clear well-defined ordering relations:

belonged BEFORE *confirmed*
belonged BEFORE *found*
found BEFORE *confirmed*
belonged BEFORE *Friday*
confirmed IS INCLUDED IN *Friday*
found IS INCLUDED IN *Friday*³

Learning algorithms handle these unlabeled edges by making incorrect assumptions, or by ignoring large parts of the temporal graph. Several models with rich temporal reasoners have been published, but since they require more connected graphs, improvement over pairwise classifiers have been minimal (Chambers and Jurafsky, 2008; Yoshikawa et al., 2009). This paper thus proposes an annotation process that builds denser graphs with formal properties that learners can rely on, such as locally complete subgraphs.

3.1 Ensuring Dense Graphs

While the ideal goal is to create a complete graph, the time it would take to hand-label $n(n-1)/2$

for accurate comparison to other corpora.

³Revealed by the previous sentence (not shown here).

edges is prohibitive. We approximate completeness by creating locally complete graphs over neighboring sentences. The resulting event graph for a document is strongly connected, but not complete. Specifically, the following edge types are included:

1. Event-Event, Event-Time, and Time-Time pairs in the same sentence
2. Event-Event, Event-Time, and Time-Time pairs between the current and next sentence
3. Event-DCT pairs for every event in the text
4. Time-DCT pairs for every time expression in the text

Our process **requires** annotators to annotate the above edge types, enforced via an annotation tool. We describe the relation set and this tool next.

3.1.1 Temporal Relations

The TimeBank corpus uses 14 relations based on the Allen interval relations. The TempEval contests have used a small set of relations (TempEval-1) and the larger set of 14 relations (TempEval-3). Published work has mirrored this trend, and different groups focus on different aspects of the semantics.

We chose a middle ground between coarse and fine-grained distinctions for annotation, settling on 6 relations: *before*, *after*, *includes*, *is included*, *simultaneous*, and *vague*. We do not adopt a more fine-grained set because we annotate pairs that are far more ambiguous than those considered in previous efforts. Decisions between relations like *before* and *immediately before* can complicate an already difficult task. The added benefit of a corpus (or working system) that makes fine-grained distinctions is also not clear. We lean toward higher annotator agreement with relations that have greater separation between their semantics⁴.

3.1.2 Enforcing Annotation

Imposing the above rules on annotators requires automated assistance. We built a new tool that reads TimeML formatted text, and computes the set of required edges. Annotators are prompted to assign a label for each edge, and skipping edges is prohibited.⁵ The tool is unique in that it includes a transitive reasoner that infers relations based on the annotator's latest annotations. For example,

⁴For instance, a relation like *starts* is a special case of *includes* if events are viewed as open intervals, and *immediately before* is a special case of *before*. We avoid this overlap and only use *includes* and *before*.

⁵Note that annotators are presented with pairs in order from document start to finish, starting with the first two events.

if event e_1 IS INCLUDED in t_1 , and t_1 BEFORE e_2 , the tool automatically labels e_1 BEFORE e_2 . The transitivity inference is run after each input label, and the human annotator cannot override the inferences. This prohibits the annotator from entering edges that break transitivity. As a result, several properties are ensured through this process: the graph (1) is a strongly connected graph, (2) is consistent with no contradictions, and (3) has all required edges labeled. These 3 properties are new to all current ordering corpora.

3.2 Annotation Guidelines

Since the annotation tool frees the annotators from the decision of *when* to label an edge, the focus is now *what* to label each edge. This section describes the guidelines for dense annotation.

The 80% confidence rule: The decision to label an edge as VAGUE instead of a defined temporal relation is critical. We adopted an 80% rule that instructed annotators to choose a specific non-vague relation if they are 80% confident that it was the writer’s intent that a reader infer that relation. By not requiring 100% confidence, we allow for alternative interpretations that conflict with the chosen edge label as long as that alternative is sufficiently unlikely. In practice, annotators had different interpretations of what constitutes 80% certainty, and this generated much discussion. We mitigated these disagreements with the following rule.

Majority annotator agreement: An edge’s label is the relation that received a majority of annotator votes, otherwise it is marked VAGUE. If a document has 2 annotators, both have to agree on the relation or it is labeled VAGUE. A document with 3 annotators requires 2 to agree. This agreement rule acts as a check to our 80% confidence rule, backing off to VAGUE when decisions are uncertain (arguably, this is the definition of VAGUE).

We also encouraged consistent labelings with guidelines inspired by Bethard and Martin (2008).

Modal and conditional events: interpreted with a *possible worlds* analysis. The core event was treated as having occurred, whether or not the text implied that it had occurred. For example,

They [EVENT expect] him to [EVENT cut] costs throughout the organization.

This event pair is ordered (expect *before* cut) since the expectation occurs before the cutting (in the

possible world where the cutting occurs). Negated events and hypotheticals are treated similarly. One assumes the event does occur, and all other events are ordered accordingly. Negated states like “is not anticipating” are interpreted as though the anticipation occurs, and surrounding events are ordered with regard to its presumed temporal span.

Aspectual Events: annotated as IS INCLUDED in their event arguments. For instance, events that describe the manner in which another event is performed are considered encompassed by the broader event. Consider the following example:

The move may [EVENT help] [EVENT prevent] Martin Ackerman from making a run at the computer-services concern.

This event pair is assigned the relation (help IS INCLUDED in prevent) because the help event is not meaningful on its own. It describes the proportion of the preventing accounted for by the move. In TimeBank, the *intentional action* class is used instead of the *aspectual* class in this case, but we still consider it covered by this guideline.

Events that attribute a property: to a person or event are interpreted to end when the entity ends. For instance, ‘the talk is nonsense’ evokes a nonsense event with an end point that coincides with the end of the talk.

Time Expressions: the words *now* and *today* were given “long now” interpretations if the words could be replaced with *nowadays* and not change the meaning of their sentences. The time’s duration starts sometime in the past and INCLUDES the DCT. If nowadays is not suitable, then the now was INCLUDED IN the DCT.

Generic Events: can be ordered with respect to each other, but must be VAGUE with respect to nearby non-generic events.

4 TimeBank-Dense: corpus statistics

We chose a subset of TimeBank documents for our new corpus: **TimeBank-Dense**. This provided an initial labeling of events and time expressions. Using the tool described above, we annotated 36 random documents with at least two annotators each. These 36 were annotated with 4 times as many relations as the entire 183 document TimeBank.

The four authors of this paper were the four annotators. All four annotated the same initial document, conflicts and disagreements were discussed,

Annotated Relation Count

BEFORE	2590	INCLUDES	836
AFTER	2104	INCLUDED IN	1060
SIMULTAN.	215	VAGUE	5910
Total Relations: 12715			

Table 2: Relation counts in TimeBank-Dense.

and guidelines were updated accordingly. The rest of the documents were then annotated independently. Document annotation was not random, but we mixed pairs of authors where time constraints allowed. Table 2 shows the relation counts in the final corpus, and Table 3 gives the annotator agreement. We show precision (holding one annotation as gold) and kappa computed on the 4 types of pairs from section 3.1. Micro-averaged precision was 65.1%, compared to TimeBank’s 77%. Kappa ranged from .56-.64, a slight drop from TimeBank’s .71.

The vague relation makes up 46% of the relations. This is the first empirical count of how many temporal relations in news articles are truly vague.

Our lower agreement is likely due to the more difficult task. Table 5 breaks down the individual disagreements. The most frequent pertained to the VAGUE relation. Practically speaking, VAGUE was applied to the final graph if either annotator chose it. This seems appropriate since a disagreement between annotators implies that the relation is vague.

The following example illustrates the difficulty of labeling edges with a VAGUE relation:

No one was **hurt**, but firefighters **ordered** the **evacuation** of nearby homes and **said** they’ll **monitor** the ground.

Both annotators chose VAGUE to label *ordered* and *said* because the order is unclear. However, they disagreed on *evacuation* with *monitor*. One chose VAGUE, but the other chose IS INCLUDED. There is a valid interpretation where a monitoring process has already begun, and continues after the evacuation. This interpretation reached 80% confidence for one annotator, but not the other. In the face of such a disagreement, the pair is labeled VAGUE.

How often do these disagreements occur? Table 4 shows the 3 sources: (1) mutual vague: annotators agree it is vague, (2) partial vague: one annotator chooses vague, but the other does not, and (3) no vague: annotators choose conflicting non-vague relations. Only 17% of these disagreements are due to hard conflicts (no vague). The released corpus includes these 3 fine-grained VAGUE relations.

Annotators	# Links	Prec	Kappa
A and B	9282	.65	.56
A and D	1605	.72	.63
B and D	279	.70	.64
C and D	1549	.65	.57

Table 3: Agreement between different annotators.

	# Vague
Mutual VAGUE	1657 (28%)
Partial VAGUE	3234 (55%)
No VAGUE	1019 (17%)

Table 4: VAGUE relation origins. Partial vague: one annotator does not choose vague. No vague: neither annotator chooses vague.

	b	a	i	ii	s	v
b	1776	22	88	37	21	192
a	17	1444	32	102	9	155
i	71	34	642	45	23	191
ii	81	76	40	826	31	230
s	12	8	25	28	147	29
v	500	441	289	356	64	1197

Table 5: Relation agreement between the two main annotators. Most disagreements involved VAGUE.

5 Conclusion

We described our annotation framework that produces corpora with formal guarantees about the annotated graph’s structure. Both the annotation tool and the new *TimeBank-Dense* corpus are publicly available.⁶ This is the first corpus with guarantees of connectedness, consistency, and a semantics for unlabeled edges. We hope to encourage a shift in the temporal ordering community to consider the entire document when making local decisions. Further work is needed to handle difficult pairs with the VAGUE relation. We look forward to evaluating new algorithms on this dense corpus.

Acknowledgments

This work was supported, in part, by the Johns Hopkins Human Language Technology Center of Excellence. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors. We also give thanks to Benjamin Van Durme for assistance and insight.

⁶<http://www.usna.edu/Users/cs/nchamber/caevo/>

References

- Steven Bethard, William J Corvey, Sara Klengenstein, and James H Martin. 2008. Building a corpus of temporal-causal structure. In *LREC*.
- Steven Bethard. 2013. Cleartk-timeml: A minimalist approach to tempeval 2013. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 10–14, Atlanta, Georgia, USA, June. Association for Computational Linguistics.
- P. Bramsen, P. Deshpande, Y.K. Lee, and R. Barzilay. 2006. Inducing temporal graphs. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 189–198. ACL.
- N. Chambers and D. Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 698–706. ACL.
- Quang Do, Wei Lu, and Dan Roth. 2012. Joint inference for event timeline construction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 677–687, Jeju Island, Korea, July. Association for Computational Linguistics.
- Oleksandr Kolomiyets, Steven Bethard, and Marie-Francine Moens. 2012. Extracting narrative timelines as temporal dependency structures. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 88–97, Jeju Island, Korea, July. Association for Computational Linguistics.
- James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. 2003. The timebank corpus. In *Corpus linguistics*, volume 2003, page 40.
- Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9, Atlanta, Georgia, USA, June. Association for Computational Linguistics.
- Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 75–80. Association for Computational Linguistics.
- Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. 2010. Semeval-2010 task 13: Tempeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 57–62. Association for Computational Linguistics.
- K. Yoshikawa, S. Riedel, M. Asahara, and Y. Matsumoto. 2009. Jointly identifying temporal relations with Markov Logic. 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 405–413. ACL.

Linguistically debatable or just plain wrong?

Barbara Plank, Dirk Hovy and Anders Søgaard

Center for Language Technology

University of Copenhagen, Denmark

Njalsgade 140, DK-2300 Copenhagen S

bplank@cst.dk, dirk@cst.dk, soegaard@hum.ku.dk

Abstract

In linguistic annotation projects, we typically develop annotation guidelines to minimize disagreement. However, in this position paper we question whether we should actually limit the disagreements between annotators, rather than embracing them. We present an empirical analysis of part-of-speech annotated data sets that suggests that disagreements are systematic across domains and to a certain extent also across languages. This points to an underlying ambiguity rather than random errors. Moreover, a quantitative analysis of tag confusions reveals that the majority of disagreements are due to linguistically debatable cases rather than annotation errors. Specifically, we show that even in the absence of annotation guidelines only 2% of annotator choices are linguistically unmotivated.

1 Introduction

In NLP, we often model annotation as if it reflected a single ground truth that was guided by an underlying linguistic theory. If this was true, the specific theory should be learnable from the annotated data. However, it is well known that there are linguistically *hard cases* (Zeman, 2010), where no theory provides a clear answer, so annotation schemes commit to more or less arbitrary decisions. For example, in parsing auxiliary verbs may head main verbs, or vice versa, and in part-of-speech (POS) tagging, possessive pronouns may belong to the category of determiners or the category of pronouns. This position paper argues that annotation projects should embrace these hard cases rather than pretend they can be unambiguously resolved. Instead of using overly specific annotation guidelines, designed to

minimize inter-annotator disagreement (Duffield et al., 2007), and adjudicating between annotators of different opinions, we should embrace systematic inter-annotator disagreements. To motivate this, we present an empirical analysis showing

1. that certain inter-annotator disagreements are systematic, and
2. that actual errors are in fact so infrequent as to be negligible, even when linguists annotate without guidelines.

The empirical analysis presented below relies on text corpora annotated with syntactic categories or parts-of-speech (POS). POS is part of most linguistic theories, but nevertheless, there are still many linguistic constructions – even very frequent ones – whose POS analysis is widely debated. The following sentences exemplify some of these hard cases that annotators frequently disagree on. Note that we do not claim that both analyses in each of these cases (1–3) are equally good, but that there is some linguistic motivation for either analysis in each case.

- | | | | | |
|-----|------|-------|---------------|----------|
| (1) | Noam | goes | out | tonight |
| | NOUN | VERB | ADP/PRT | ADV/NOUN |
| (2) | Noam | likes | social | media |
| | NOUN | VERB | ADJ/NOUN | NOUN |
| (3) | Noam | likes | his | car |
| | NOUN | VERB | DET/PRON | NOUN |

To substantiate our claims, we first compare the distribution of inter-annotator disagreements across domains and languages, showing that most disagreements are systematic (Section 2). This suggests that most annotation differences derive from hard cases, rather than random errors.

We then collect a corpus of such disagreements and have experts mark which ones are due to actual annotation *errors*, and which ones reflect linguistically hard cases (Section 3). The results show that the majority of disagreements are due

to hard cases, and only about 20% of conflicting annotations are actual errors. This suggests that inter-annotator agreement scores often hide the fact that the vast majority of annotations are actually linguistically motivated. In our case, less than 2% of the overall annotations are linguistically unmotivated.

Finally, in Section 4, we present an experiment trying to learn a model to distinguish between hard cases and annotation errors.

2 Annotator disagreements across domains and languages

In this study, we had between 2-10 individual annotators with degrees in linguistics annotate different kinds of English text with POS tags, e.g., newswire text (PTB WSJ Section 00), transcripts of spoken language (from a database containing transcripts of conversations, Talkbank¹), as well as Twitter posts. Annotators were specifically *not* presented with guidelines that would help them resolve hard cases. Moreover, we compare professional annotation to that of lay people. We instructed annotators to use the 12 universal POS tags of Petrov et al. (2012). We did so in order to make comparison between existing data sets possible. Moreover, this allows us to focus on really hard cases, as any debatable case in the coarse-grained tag set is necessarily also part of the finer-grained tag set.² For each domain, we collected exactly 500 doubly-annotated sentences/tweets. Besides these English data sets, we also obtained doubly-annotated POS data from the French Social Media Bank project (Seddah et al., 2012).³ All data sets, except the French one, are publicly available at <http://lowlands.ku.dk/>.

We present disagreements as Hinton diagrams in Figure 1a–c. Note that the spoken language data does not include punctuation. The correlations between the disagreements are highly significant, with Spearman coefficients ranging from 0.644

¹<http://talkbank.org/>

²Experiments with variation n -grams on WSJ (Dickinson and Meurers, 2003) and the French data lead us to estimate that the fine-to-coarse mapping of POS tags disregards about 20% of observed tag-pair confusion types, most of which relate to fine-grained verb and noun distinctions, e.g. past participle versus past in “[...] criminal lawyers speculated/VBD vs. VBN that [...]”.

³We mapped POS tags into the universal POS tags using the mappings available here: <https://code.google.com/p/universal-pos-tags/>

(spoken and WSJ) to 0.869 (spoken and Twitter). Kendall’s τ ranges from 0.498 (Twitter and WSJ) to 0.659 (spoken and Twitter). All diagrams have a vaguely “dagger”-like shape, with the blade going down the diagonal from top left to bottom right, and a slightly curved “hilt” across the counter-diagonal, ending in the more pronounced ADP/PRT confusion cells.

Disagreements are very similar across all three domains. In particular, adpositions (ADP) are confused with particles (PRT) (as in the case of “*get out*”); adjectives (ADJ) are confused with nouns (as in “*stone lion*”); pronouns (PRON) are confused with determiners (DET) (“*my house*”); numerals are confused with adjectives, determiners, and nouns (“*2nd time*”); and adjectives are confused with adverbs (ADV) (“*see you later*”). In Twitter, the X category is often confused with punctuations, e.g., when annotating punctuation acting as discourse continuation marker.

Our analyses show that a) experts disagree on the known hard cases when freely annotating text, and b) that these disagreements are the same across text types. More surprisingly, though, we also find that, as discussed next, c) roughly the same disagreements are also observed when comparing the linguistic intuitions of lay people.

More specifically, we had lay annotators on the crowdsourcing platform Crowdfunder re-annotate the training section of Gimpel et al. (2011). They collected five annotations per word. Only annotators that had answered correctly on 4 gold items (randomly chosen from a set of 20 gold items provided by the authors) were allowed to submit annotations. In total, 177 individual annotators supplied answers. We paid annotators a reward of \$0.05 for 10 items. The full data set contains 14,619 items and is described in further detail in Hovy et al. (2014). Annotators were satisfied with the task (4.5 on a scale from 1 to 5) and felt that instructions were clear (4.4/5), and the pay reasonable (4.1/5). The crowdsourced annotations aggregated using majority voting agree with the expert annotations in 79.54% of the cases. If we pre-filter the data via Wiktionary and use an item-response model (Hovy et al., 2013) rather than majority voting, the agreement rises to 80.58%.

Figure 2 presents the Hinton diagram of the disagreements of lay people. Disagreements are very similar to the disagreements between expert annotators, especially on Twitter data (Figure 1b).

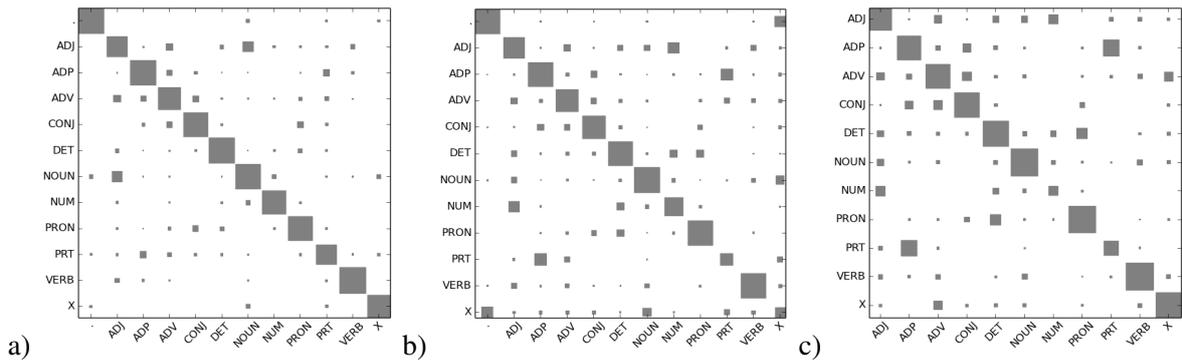


Figure 1: Hinton diagrams of inter-annotator disagreement on (a) excerpt from WSJ (Marcus et al., 1993), (b) random Twitter sample, and (c) pre-transcribed spoken language excerpts from talkbank.org

One difference is that lay people do not confuse numerals very often, probably because they rely more on orthographic cues than on distributional evidence. The disagreements are still strongly correlated with the ones observed with expert annotators, but at a slightly lower coefficient (with a Spearman’s ρ of 0.493 and Kendall’s τ of 0.366 for WSJ).

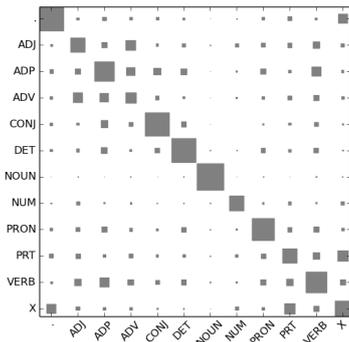


Figure 2: Disagreement between lay annotators

Lastly, we compare the disagreements of annotators on a French social media data set (Seddah et al., 2012), which we mapped to the universal POS tag set. Again, we see the familiar dagger shape. The Spearman coefficient with English Twitter is 0.288; Kendall’s τ is 0.204. While the correlation is weaker across languages than across domains, it remains statistically significant ($p < 0.001$).

3 Hard cases and annotation errors

In the previous section, we demonstrated that some disagreements are consistent across domains and languages. We noted earlier, though, that disagreements can arise both from hard cases and from annotation errors. This can explain some

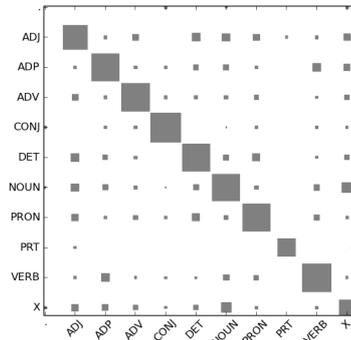


Figure 3: Disagreement on French social media

of the variation. In this section, we investigate what happens if we weed out obvious errors by detecting annotation inconsistencies across a corpus. The disagreements that remain are the truly hard cases.

We use a modified version of the a priori algorithm introduced in Dickinson and Meurers (2003) to identify annotation inconsistencies. It works by collecting “variation n -grams”, i.e. the longest sequence of words (n -gram) in a corpus that has been observed with a token being tagged differently in another occurrence of the same n -gram in the same corpus. The algorithm starts off by looking for unigrams and expands them until no longer n -grams are found.

For each variation n -gram that we found in WSJ-00, i.e. a word in various contexts and the possible tags associated with it, we present annotators with the cross product of contexts and tags. Essentially, we ask for a binary decision: Is the tag plausible for the given context?

We used 3 annotators with PhD degrees in linguistics. In total, our data set contains 880 items,

i.e. 440 annotated confusion tag pairs. The raw agreement was 86%. Figure 4 shows how truly hard cases are distributed over tag pairs (dark gray bars), as well as the proportion of confusions with respect to a given tag pair that are truly hard cases (light gray bars). The figure shows, for instance, that the variation n -gram regarding ADP-ADV is the second most frequent one (dark gray), and approximately 70% of ADP-ADV disagreements are linguistically hard cases (light gray). NOUN-PRON disagreements are always linguistically debatable cases, while they are less frequent.

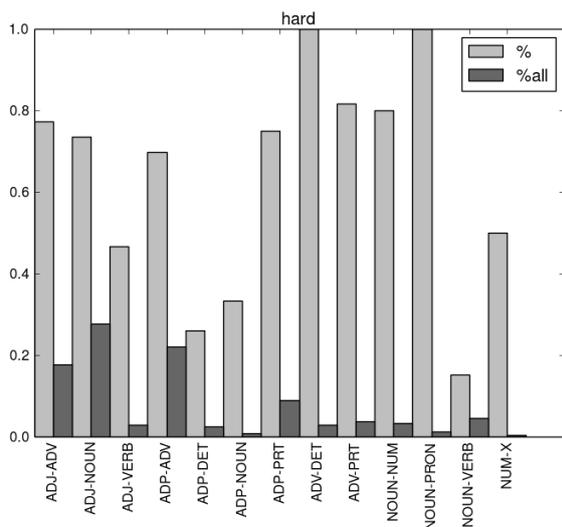


Figure 4: Relative frequency of hard cases

A survey of hard cases. To further test the idea of there being truly hard cases that probably cannot be resolved by linguistic theory, we presented nine linguistics faculty members with 10 of the above examples and asked them to pick their favorite analyses. In 8/10 cases, the faculty members disagreed on the right analysis.

4 Learning to detect annotation errors

In this section, we examine whether we can learn a classifier to distinguish between hard cases and annotation errors. In order to do so, we train a classifier on the annotated data set containing 440 tag-confusion pairs by relying only on surface form features. If we *balance* the data set and perform 3-fold cross-validation, a L2-regularized logistic regression (L2-LR) model achieves an f_1 -score for detecting errors at 70% (cf. Table 1), which is above average, but not very impressive.

The two classes are apparently not easily separable using surface form features, as illustrated in

f_1	HARD CASES	ERRORS
L2-LR	73%(71-77)	70%(65-75)
NN	76%(76-77)	71%(68-72)

Table 1: Classification results

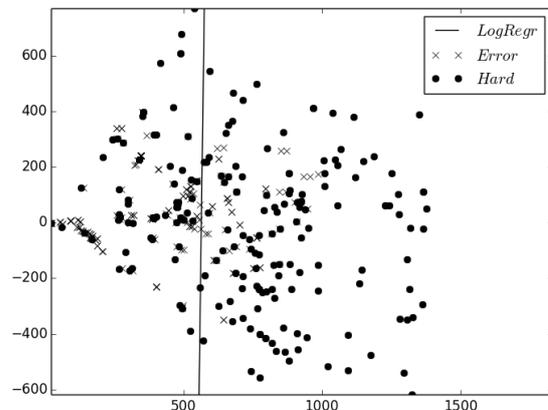


Figure 5: Hard cases and errors in 2d-PCA

the two-dimensional plot in Figure 5, obtained using PCA. The logistic regression decision boundary is plotted as a solid, black line. This is probably also why the nearest neighbor (NN) classifier does slightly better, but again, performance is rather low. While other features may reveal that the problem is in fact learnable, our initial experiments lead us to conclude that, given the low ratio of errors over truly hard cases, learning to detect errors is often not worthwhile.

5 Related work

Juergens (2014) presents work on detecting linguistically hard cases in the context of word sense annotations, e.g., cases where expert annotators will disagree, as well as differentiating between underspecified, overspecified and metaphoric cases. This work is similar to ours in spirit, but considers a very different task. While we also quantify the proportion of hard cases and present an analysis of these cases, we also show that disagreements are systematic.

Our work also relates to work on automatically correcting expert annotations for inconsistencies (Dickinson and Meurers, 2003). This work is very different in spirit from our work, but shares an interest in reconsidering expert annotations, and we made use of their mining algorithm here. There has also been recent work on adjudicat-

ing noisy crowdsourced annotations (Dawid and Skene, 1979; Smyth et al., 1995; Carpenter, 2008; Whitehill et al., 2009; Welinder et al., 2010; Yan et al., 2010; Raykar and Yu, 2012; Hovy et al., 2013). Again, their objective is orthogonal to ours, namely to collapse multiple annotations into a gold standard rather than embracing disagreements.

Finally, Plank et al. (2014) use small samples of doubly-annotated POS data to estimate annotator reliability and show how those metrics can be implemented in the loss function when inducing POS taggers to reflect confidence we can put in annotations. They show that not biasing the theory towards a single annotator but using a cost-sensitive learning scheme makes POS taggers more robust and more applicable for downstream tasks.

6 Conclusion

In this paper, we show that disagreements between professional or lay annotators are systematic and consistent across domains and some of them are systematic also across languages. In addition, we present an empirical analysis of POS annotations showing that the vast majority of inter-annotator disagreements are competing, but valid, linguistic interpretations. We propose to embrace such disagreements rather than using annotation guidelines to optimize inter-annotator agreement, which would bias our models in favor of some linguistic theory.

Acknowledgements

We would like to thank the anonymous reviewers for their feedback, as well as Djamé Seddah for the French data. This research is funded by the ERC Starting Grant LOWLANDS No. 313695.

References

Bob Carpenter. 2008. Multilevel Bayesian models of categorical data annotation. Technical report, LingPipe.

A. Philip Dawid and Allan M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, pages 20–28.

Markus Dickinson and Detmar Meurers. 2003. Detecting errors in part-of-speech annotation. In *EACL*.

Cecily Duffield, Jena Hwang, Susan Brown, Dmitriy Dligach, Sarah Vieweg, Jenny Davis, and Martha

Palmer. 2007. Criteria for the manual grouping of verb senses. In *LAW*.

- Kevin Gimpel, Nathan Schneider, Brendan O’Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In *ACL*.
- Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *NAACL*.
- Dirk Hovy, Barbara Plank, and Anders Søgaard. 2014. Experiments with crowdsourced re-annotation of a POS tagging data set. In *ACL*.
- David Juergens. 2014. An analysis of ambiguity in word sense annotations. In *LREC*.
- Mitchell Marcus, Mary Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.
- Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *LREC*.
- Barbara Plank, Dirk Hovy, and Anders Søgaard. 2014. Learning part-of-speech taggers with inter-annotator agreement loss. In *EACL*.
- Vikas C. Raykar and Shipeng Yu. 2012. Eliminating Spammers and Ranking Annotators for Crowdsourced Labeling Tasks. *Journal of Machine Learning Research*, 13:491–518.
- Djamé Seddah, Benoit Sagot, Marie Candito, Virginie Moulleron, and Vanessa Combet. 2012. The French Social Media Bank: a treebank of noisy user generated content. In *COLING*.
- Padhraic Smyth, Usama Fayyad, Mike Burl, Pietro Perona, and Pierre Baldi. 1995. Inferring ground truth from subjective labelling of Venus images. In *NIPS*.
- Peter Welinder, Steve Branson, Serge Belongie, and Pietro Perona. 2010. The multidimensional wisdom of crowds. In *NIPS*.
- Jacob Whitehill, Paul Ruvolo, Tingfan Wu, Jacob Bergsma, and Javier Movellan. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *NIPS*.
- Yan Yan, Rómer Rosales, Glenn Fung, Mark Schmidt, Gerardo Hermosillo, Luca Bogoni, Linda Moy, and Jennifer Dy. 2010. Modeling annotator expertise: Learning when everybody knows a bit of something. In *AIStats*.
- Daniel Zeman. 2010. Hard problems of tagset conversion. In *Proceedings of the Second International Conference on Global Interoperability for Language Resources*.

Humans Require Context to Infer Ironic Intent (so Computers Probably do, too)

Byron C. Wallace, Do Kook Choe, Laura Kertz and Eugene Charniak
Brown University

{byron_wallace, do_kook_choe, laura_kertz, eugene_charniak}@brown.edu

Abstract

Automatically detecting verbal irony (roughly, sarcasm) is a challenging task because ironists say something other than – and often opposite to – what they actually mean. Discerning ironic intent exclusively from the words and syntax comprising texts (e.g., tweets, forum posts) is therefore not always possible: additional contextual information about the speaker and/or the topic at hand is often necessary. We introduce a new corpus that provides empirical evidence for this claim. We show that annotators frequently require context to make judgements concerning ironic intent, and that machine learning approaches tend to misclassify those same comments for which annotators required additional context.

1 Introduction & Motivation

This work concerns the task of detecting verbal irony online. Our principal argument is that simple bag-of-words based text classification models – which, when coupled with sufficient data, have proven to be extremely successful for many natural language processing tasks (Halevy et al., 2009) – are inadequate for irony detection. In this paper we provide empirical evidence that *context* is often necessary to recognize ironic intent.

This is consistent with the large body of pragmatics/linguistics literature on irony and its usage, which has emphasized the role that context plays in recognizing and decoding ironic utterances (Grice, 1975; Clark and Gerrig, 1984; Sperber and Wilson, 1981). But existing work on automatic irony detection – reviewed in Section 2 – has not explicitly attempted to operationalize such theories, and has instead relied on features

(mostly word counts) intrinsic to the texts that are to be classified as ironic. These approaches have achieved some success, but necessarily face an upper-bound: the *exact same sentence* can be both intended ironically and unironically, depending on the context (including the speaker and the topic at hand). Only obvious verbal ironies will be recognizable from intrinsic features alone.

Here we provide empirical evidence for the above claims. We also introduce a new annotated corpus that will allow researchers to build models that augment existing approaches to irony detection with contextual information regarding the text (utterance) to be classified and its author. Briefly, our contributions are summarized as follows.

- We introduce the first version of the *reddit irony corpus*, composed of annotated comments from the social news website reddit. Each sentence in every comment in this corpus has been labeled by three independent annotators as having been intended by the author ironically or not. This dataset is publicly available.¹
- We provide empirical evidence that human annotators consistently rely on contextual information to make ironic/unironic sentence judgements.
- We show that the standard ‘bag-of-words’ approach to text classification fails to accurately judge ironic intent on those cases for which humans required additional context. This suggests that, as humans require context to make their judgements for this task, so too do computers.

Our hope is that these observations and this dataset will spur innovative new research on methods for verbal irony detection.

¹<https://github.com/bwallace/ACL-2014-irony>

2 Previous Work

There has recently been a flurry of interesting work on automatic irony detection (Tepperman et al., 2006; Davidov et al., 2010; Carvalho et al., 2009; Burfoot and Baldwin, 2009; Tsur et al., 2010; González-Ibáñez et al., 2011; Filatova, 2012; Reyes et al., 2012; Lukin and Walker, 2013; Riloff et al., 2013). In these works, verbal irony detection has mostly been treated as a standard text classification task, though with some innovative approaches specific to detecting irony.

The most common data source used to experiment with irony detection systems has been Twitter (Reyes et al., 2012; González-Ibáñez et al., 2011; Davidov et al., 2010), though Amazon product reviews have been used experimentally as well (Tsur et al., 2010; Davidov et al., 2010; Reyes et al., 2012; Filatova, 2012). Walker et al. (2012) also recently introduced the Internet Argument Corpus (IAC), which includes a *sarcasm* label (among others).

Some of the findings from these previous efforts have squared with intuition: e.g., overzealous punctuation (as in “great idea!!!!”) is indicative of ironic intent (Carvalho et al., 2009). Other works have proposed novel approaches specifically for irony detection: Davidov et al. (2010), for example, proposed a semi-supervised approach in which they look for sentence *templates* indicative of irony. Elsewhere, Riloff et al. (2013) proposed a method that exploits contrasting sentiment in the same utterance to detect irony.

To our knowledge, however, no previous work on irony detection has attempted to leverage *contextual* information regarding the author or speaker (external to the utterance). But this is necessary in some cases, however. For example, in the case of Amazon product reviews, knowing the kinds of books that an individual typically likes might inform our judgement: someone who tends to read and review Dostoevsky is probably being ironic if she writes a glowing review of *Twilight*. Of course, many people genuinely do enjoy *Twilight* and so if the review is written subtly it will likely be difficult to discern the author’s intent without this background. In the case of Twitter, it is likely to be difficult to classify utterances without considering the contextualizing exchange of tweets (i.e., the conversation) to which they belong.

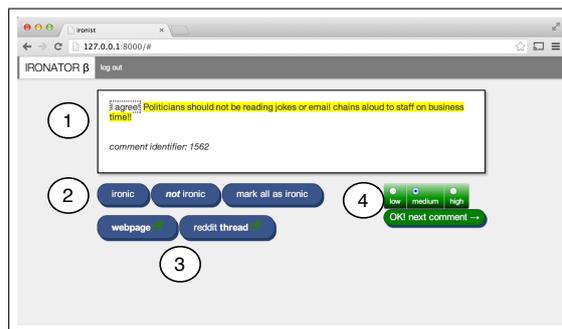


Figure 1: The web-based tool used by our annotators to label reddit comments. Enumerated interface elements are described as follows: **1** the text of the comment to be annotated – sentences marked as *ironic* are highlighted; **2** buttons to label sentences as *ironic* or *unironic*; **3** buttons to request additional *context* (the embedding discussion thread or associated webpage – see Section 3.2); **4** radio button to provide *confidence* in comment labels (*low*, *medium* or *high*).

3 Introducing the reddit Irony Dataset

Here we introduce the first version (β 1.0) of our irony corpus. Reddit (<http://reddit.com>) is a social-news website to which news stories (and other links) are posted, voted on and commented upon. The forum component of reddit is extremely active: popular posts often have well into 1000’s of user comments. Reddit comprises ‘sub-reddits’, which focus on specific topics. For example, <http://reddit.com/r/politics> features articles (and hence comments) centered around political news. The current version of the corpus is available at: <https://github.com/bwallace/ACL-2014-irony>. Data collection and annotation is ongoing, so we will continue to release new (larger) versions of the corpus in the future. The present version comprises 3,020 annotated comments scraped from the six subreddits enumerated in Table 1. These comments in turn comprise a total of 10,401 labeled sentences.²

3.1 Annotation Process

Three university undergraduates independently annotated each sentence in the corpus. More specifically, annotators have provided binary ‘labels’ for each sentence indicating whether or not they (the annotator) believe it was intended by the author ironically (or not). This annotation was provided via a custom-built browser-based annotation tool, shown in Figure 1.

We intentionally did not provide much guidance to annotators regarding the criteria for what

²We performed naïve ‘segmentation’ of comments based on punctuation.

sub-reddit (URL)	description	number of labeled comments
politics (r/politics)	Political news and editorials; focus on the US.	873
conservative (r/conservative)	A community for political conservatives.	573
progressive (r/progressive)	A community for political progressives (liberals).	543
atheism (r/atheism)	A community for non-believers.	442
Christianity (r/Christianity)	News and viewpoints on the Christian faith.	312
technology (r/technology)	Technology news and commentary.	277

Table 1: The six sub-reddits that we have downloaded comments from and the corresponding number of comments for which we have acquired annotations in this β version of the corpus. Note that we acquired labels at the *sentence* level, whereas the counts above reflect *comments*, all of which contain at least one sentence.

constitutes an ‘ironic’ statement, for two reasons. First, verbal irony is a notoriously slippery concept (Gibbs and Colston, 2007) and coming up with an operational definition to be consistently applied is non-trivial. Second, we were interested in assessing the extent of natural agreement between annotators for this task. The raw average agreement between all annotators on all sentences is 0.844. Average pairwise Cohen’s Kappa (Cohen, 1960) is 0.341, suggesting fair to moderate agreement (Viera and Garrett, 2005), as we might expect for a subjective task like this one.

3.2 Context

Reddit is a good corpus for the irony detection task in part because it provides a natural practical realization of the otherwise ill-defined *context* for comments. In particular, each comment is associated with a specific user (the author), and we can view their previous comments. Moreover, comments are embedded within discussion *threads* that pertain to the (usually external) content linked to in the corresponding submission (see Figure 2). These pieces of information (previous comments by the same user, the external link of the embedding reddit thread, and the other comments in this thread) constitute our context. All of this is readily accessible. Labelers can opt to request these pieces of context via the annotation tool, and we record when they do so.

Consider the following example comment taken from our dataset: “Great idea on the talkathon Cruz. Really made the republicans look like the sane ones.” Did the author intend this statement ironically, or was this a subtle dig on Senator Ted Cruz? Without additional context it is difficult to know. And indeed, all three annotators requested additional context for this comment. This context at first suggests that the comment may have been intended literally: it was posted in the r/conservative subreddit (Ted Cruz is a conservative senator). But if we peruse the author’s com-



Figure 2: An illustrative reddit comment (highlighted). The title (“Virginia Republican ...”) links to an article, providing one example of contextualizing content. The conversational thread in which this comment is embedded provides additional context. The comment in question was presumably intended ironically, though without the aforementioned context this would be difficult to conclude with any certainty.

ment history, we see that he or she repeatedly derides Senator Cruz (e.g., writing “Ted Cruz is no Ronald Reagan. They aren’t even close.”). From this contextual information, then, we can reasonably assume that the comment was intended ironically (and all three annotators did so after assessing the available contextual information).

4 Humans Need Context to Infer Irony

We explore the extent to which human annotators rely on contextual information to decide whether or not sentences were intended ironically. Recall that our annotation tool allows labelers to request additional context if they cannot make a decision based on the comment text alone (Figure 1). On average, annotators requested additional context for 30% of comments (range across annotators of 12% to 56%). As shown in Figure 3, annotators are consistently more confident once they have consulted this information.

We tested for a correlation between these requests for context and the final decisions regarding whether comments contain at least one ironic sentence. We denote the probability of at least one annotator requesting additional context for comment i by $P(C_i)$. We then model the probability of this event as a linear function of whether or not

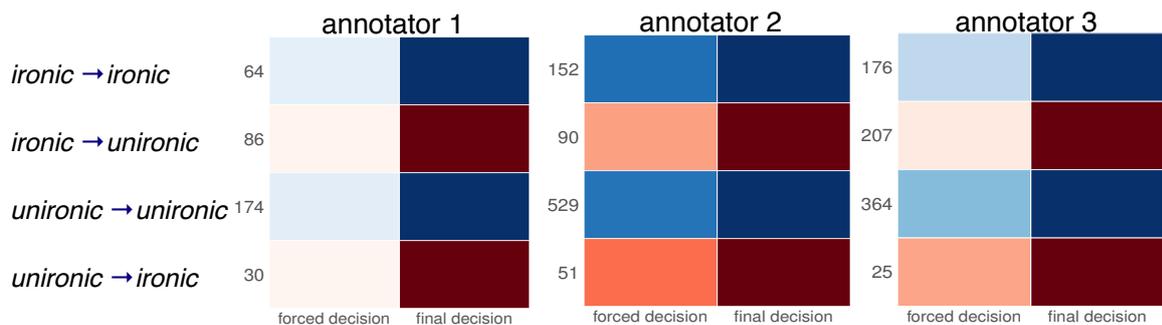


Figure 3: This plot illustrates the effect of viewing contextual information for three annotators (one table for each annotator). For all comments for which these annotators requested context, we show *forced* (before viewing the requested contextual content) and *final* (after) decisions regarding perceived ironic intent on behalf of the author. Each row shows one of four possible decision sequences (e.g., a judgement of *ironic* prior to seeing context and *unironic* after). Numbers correspond to counts of these sequences for each annotator (e.g., the first annotator changed their mind from *ironic* to *unironic* 86 times). Cases that involve the annotator changing his or her mind are shown in red; those in which the annotator stuck with their initial judgement are shown in blue. Color intensity is proportional to the average confidence judgements the annotator provided: these are uniformly stronger after they have consulted contextualizing information. Note also that the context frequently results in annotators changing their judgement.

any annotator labeled any sentence in comment i as ironic. We code this via the indicator variable \mathcal{I}_i which is 1 when comment i has been deemed to contain an ironic sentence (by any of the three annotators) and 0 otherwise.

$$\text{logit}\{P(C_i)\} = \beta_0 + \beta_1 \mathcal{I}_i \quad (1)$$

We used the regression model shown in Equation 1, where β_0 is an intercept and β_1 captures the correlation between requests for context for a given comment and its ultimately being deemed to contain at least one ironic sentence. We fit this model to the annotated corpus, and found a significant correlation: $\hat{\beta}_1 = 1.508$ with a 95% confidence interval of (1.326, 1.690); $p < 0.001$.

In other words, annotators request context significantly more frequently for those comments that (are ultimately deemed to) contain an ironic sentence. This would suggest that the words and punctuation comprising online comments alone are not sufficient to distinguish ironic from unironic comments. Despite this, most machine learning based approaches to irony detection have relied nearly exclusively on such intrinsic features.

5 Machines Probably do, too

We show that the misclassifications (with respect to whether comments contain irony or not) made by a standard text classification model significantly correlate with those comments for which human annotators requested additional context. This provides evidence that bag-of-words approaches are insufficient for the general task of

irony detection: more context is necessary.

We implemented a baseline classification approach using vanilla token count features (binary bag-of-words). We removed stop-words and limited the vocabulary to the 50,000 most frequently occurring unigrams and bigrams. We added additional binary features coding for the presence of punctuational features, such as exclamation points, emoticons (for example, ‘;’) and question marks: previous work (Davidov et al., 2010; Carvalho et al., 2009) has found that these are good indicators of ironic intent.

For our predictive model, we used a linear-kernel SVM (tuning the C parameter via grid-search over the training dataset to maximize F1 score). We performed five-fold cross-validation, recording the predictions \hat{y}_i for each (held-out) comment i . Average F1 score over the five-folds was 0.383 with range (0.330, 0.412); mean recall was 0.496 (0.446, 0.548) and average precision was 0.315 (0.261, 0.380). The five most predictive tokens were: *!*, *yeah*, *guys*, *oh* and *shocked*. This represents reasonable performance (with intuitive predictive tokens); but obviously there is quite a bit of room for improvement.³

We now explore empirically whether these misclassifications are made on the same comments for which annotators requested context. To this end, we introduce a variable \mathcal{M}_i for each comment i such that $\mathcal{M}_i = 1$ if $\hat{y}_i \neq y_i$, i.e., \mathcal{M}_i is an in-

³Some of the recently proposed strategies mentioned in Section 2 may improve performance here, but none of these address the fundamental issue of *context*.

indicator variable that encodes whether or not the classifier misclassified comment i . We then ran a second regression in which the output variable was the logit-transformed probability of the model misclassifying comment i , i.e., $P(\mathcal{M}_i)$. Here we are interested in the correlation of the event that one or more annotators requested additional context for comment i (denoted by \mathcal{C}_i) and model misclassifications (adjusting for the comment’s true label). Formally:

$$\text{logit}\{P(\mathcal{M}_i)\} = \theta_0 + \theta_1\mathcal{I}_i + \theta_2\mathcal{C}_i \quad (2)$$

Fitting this to the data, we estimated $\hat{\theta}_2 = 0.971$ with a 95% CI of (0.810, 1.133); $p < 0.001$. Put another way, the model makes mistakes on those comments for which annotators requested additional context (even after accounting for the annotator designation of comments).

6 Conclusions and Future Directions

We have described a new (publicly available) corpus for the task of verbal irony detection. The data comprises comments scraped from the social news website reddit. We recorded confidence judgements and requests for contextualizing information for each comment during annotation. We analyzed this corpus to provide empirical evidence that annotators quite often require context beyond the comment under consideration to discern irony; especially for those comments ultimately deemed as being intended ironically. We demonstrated that a standard token-based machine learning approach misclassified many of the same comments for which annotators tend to request context.

We have shown that annotators rely on contextual cues (in addition to word and grammatical features) to discern irony and argued that this implies computers should, too. The obvious next step is to develop new machine learning models that exploit the contextual information available in the corpus we have curated (e.g., previous comments by the same user, the thread topic).

7 Acknowledgement

This work was made possible by the Army Research Office (ARO), grant #64481-MA.

References

C Burfoot and T Baldwin. 2009. Automatic satire detection: are you having a laugh? In *ACL-IJCNLP*, pages 161–164. ACL.

P Carvalho, L Sarmiento, MJ Silva, and E de Oliveira. 2009. Clues for detecting irony in user-generated

contents: oh...!! it’s so easy;-). In *CIKM workshop on Topic-sentiment analysis for mass opinion*, pages 53–56. ACM.

- HH Clark and RJ Gerrig. 1984. On the pretense theory of irony. *Journal of Experimental Psychology*, 113:121–126.
- J Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46.
- D Davidov, O Tsur, and A Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twitter and amazon. pages 107–116.
- E Filatova. 2012. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *LREC*, volume 12, pages 392–398.
- RW Gibbs and HL Colston. 2007. *Irony in language and thought: a cognitive science reader*. Lawrence Erlbaum.
- R González-Ibáñez, S Muresan, and N Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *ACL*, volume 2, pages 581–586. Citeseer.
- HP Grice. 1975. Logic and conversation. 1975, pages 41–58.
- A Halevy, P Norvig, and F Pereira. 2009. The unreasonable effectiveness of data. *Intelligent Systems, IEEE*, 24(2):8–12.
- S Lukin and M Walker. 2013. Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue. *NAACL*, pages 30–40.
- A Reyes, P Rosso, and T Veale. 2012. A multidimensional approach for detecting irony in twitter. *LREC*, pages 1–30.
- E Riloff, A Qadir, P Surve, LD Silva, N Gilbert, and R Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *EMNLP*, pages 704–714.
- D Sperber and D Wilson. 1981. Irony and the use-mention distinction. 1981.
- J Tepperman, D Traum, and S Narayanan. 2006. “Yeah Right”: Sarcasm Recognition for Spoken Dialogue Systems.
- O Tsur, D Davidov, and A Rappoport. 2010. ICWSM—a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *AAAI Conference on Weblogs and Social Media*.
- AJ Viera and JM Garrett. 2005. Understanding interobserver agreement: the kappa statistic. *Family Medicine*, 37(5):360–363.
- MA Walker, JEF Tree, P Anand, R Abbott, and J King. 2012. A corpus for research on deliberation and debate. In *LREC*, pages 812–817.

Automatic prediction of aspectual class of verbs in context

Annemarie Friedrich and Alexis Palmer

Department of Computational Linguistics

Saarland University, Saarbrücken, Germany

{afried, apalmer}@coli.uni-saarland.de

Abstract

This paper describes a new approach to predicting the aspectual class of verbs in context, i.e., whether a verb is used in a stative or dynamic sense. We identify two challenging cases of this problem: when the verb is unseen in training data, and when the verb is ambiguous for aspectual class. A semi-supervised approach using linguistically-motivated features and a novel set of distributional features based on representative verb types allows us to predict classes accurately, even for unseen verbs. Many frequent verbs can be either stative or dynamic in different contexts, which has not been modeled by previous work; we use contextual features to resolve this ambiguity. In addition, we introduce two new datasets of clauses marked for aspectual class.

1 Introduction

In this work, we focus on the automatic prediction of whether a verb in context is used in a *stative* or in a *dynamic* sense, the most fundamental distinction in all taxonomies of *aspectual class*. The aspectual class of a discourse's finite verbs is an important factor in conveying and interpreting temporal structure (Moens and Steedman, 1988; Dorr, 1992; Klavans and Chodorow, 1992); others are tense, grammatical aspect, mood and whether the utterance represents an event as completed. More accurate temporal information processing is expected to be beneficial for a variety of natural language processing tasks (Costa and Branco, 2012; UzZaman et al., 2013).

While most verbs have one predominant interpretation, others are more flexible for aspectual class and can occur as either stative (1) or dynamic (2) depending on the context. There are also cases that allow for both readings, such as (3).

(1) *The liquid **fills** the container. (stative)*

(2) *The pool slowly **filled** with water. (dynamic)*

(3) *Your soul was made to be **filled** with God Himself. (both)* (Brown corpus, religion)

Cases like (3) do not imply that there is a third class, but rather that two interpretations are available for the sentence, of which usually one will be chosen by a reader.

Following Siegel and McKeown (2000), we aim to automatically classify clauses for *fundamental aspectual class*, a function of the main verb and a select group of complements, which may differ per verb (Siegel and McKeown, 2000; Siegel, 1998b). This corresponds to the aspectual class of the clause's main verb when ignoring any aspectual markers or transformations. For example, English sentences with perfect tense are usually considered to introduce states to the discourse (Smith, 1991; Katz, 2003), but we are interested in the aspectual class before this transformation takes place. The clause *John has kissed Mary* introduces a state, but the fundamental aspectual class of the 'tenseless' clause *John kiss Mary* is dynamic.

In contrast to Siegel and McKeown (2000), we do not conduct the task of predicting aspectual class solely at the type level, as such an approach ignores the minority class of ambiguous verbs. Instead we predict the aspectual class of verbs in the context of their arguments and modifiers. We show that this method works better than using only type-based features, especially for verbs with ambiguous aspectual class. In addition, we show that type-based features, including novel distributional features based on representative verbs, accurately predict predominant aspectual class for unseen verb types. Our work differs from prior work in that we treat the problem as a three-way classification task, predicting DYNAMIC, STATIVE or BOTH as the aspectual class of a verb in context.

2 Related work

Aspectual class is well treated in the linguistic literature (Vendler, 1957; Dowty, 1979; Smith, 1991, for example). Our notion of the stative/dynamic distinction corresponds to Bach’s (1986) distinction between states and non-states; to states versus occurrences (events and processes) according to Mourelatos (1978); and to Vendler’s (1957) distinction between states and the other three classes (activities, achievements, accomplishments).

Early studies on the computational modeling of aspectual class (Nakhimovsky, 1988; Passonneau, 1988; Brent, 1991; Klavans and Chodorow, 1992) laid foundations for a cluster of papers published over a decade ago (Siegel and McKeown, 2000; Siegel, 1998b; Siegel, 1998a). Since then, it has mostly been treated as a subtask within temporal reasoning, such as in efforts related to TimeBank (Pustejovsky et al., 2003) and the TempEval challenges (Verhagen et al., 2007; Verhagen et al., 2010; UzZaman et al., 2013), where top-performing systems (Jung and Stent, 2013; Bethard, 2013; Chambers, 2013) use corpus-based features, WordNet synsets, parse paths and features from typed dependencies to classify events as a joint task with determining the event’s span. Costa and Branco (2012) explore the usefulness of a wider range of explicitly aspectual features for temporal relation classification.

Siegel and McKeown (2000) present the most extensive study of predicting aspectual class, which is the main inspiration for this work. While all of their linguistically motivated features (see section 4.1) are type-based, they train on and evaluate over labeled verbs in context. Their data set taken from medical discharge summaries comprises 1500 clauses containing main verbs other than *be* and *have* which are marked for aspectual class. Their model fails to outperform a baseline of memorizing the most frequent class of a verb type, and they present an experiment testing on unseen verb types only for the related task of classifying completedness of events. We replicate their method using publicly available software, create a similar but larger corpus,¹ and show that it is indeed possible to predict the aspectual class of unseen verbs. Siegel (1998a) investigates a classification method for the verb *have* in context; in-

¹Direct comparison on their data is not possible; feature values for the verbs studied are available, but full texts and the English Slot Grammar parser (McCord, 1990) are not.

genre	COMPLETE		W/O <i>have/be/none</i>	
	clauses	κ	clauses	κ
jokes	3462	0.85	2660	0.77
letters	1848	0.71	1444	0.62
news	2565	0.79	2075	0.69
all	7875	0.80	6161	0.70

Table 1: **Asp-MASC**: Cohen’s observed un-weighted κ .

	DYNAMIC	STATIVE	BOTH
DYNAMIC	4464	164	9
STATIVE	434	1056	29
BOTH	5	0	0

Table 2: **Asp-MASC**: confusion matrix for two annotators, without *have/be/none* clauses, κ is 0.7.

spired by this work, our present work goes one step further and uses a larger set of *instance-based* contextual features to perform experiments on a set of 20 verbs. To the best of our knowledge, there is no previous work comprehensively addressing aspectual classification of verbs in context.

3 Data

Verb type seed sets Using the **LCS Database** (Dorr, 2001), we identify sets of verb types whose senses are only stative (188 verbs, e.g. *belong, cost, possess*), only dynamic (3760 verbs, e.g. *alter, knock, resign*), or mixed (215 verbs, e.g. *fill, stand, take*), following a procedure described by Dorr and Olsen (1997).

Asp-MASC The Asp-MASC corpus consists of 7875 clauses from the letters, news and jokes sections of MASC (Ide et al., 2010), each labeled by two annotators for the aspectual class of the main verb.² Texts were segmented into clauses using SPADE (Soricut and Marcu, 2003) with some heuristic post-processing. We parse the corpus using the Stanford dependency parser (De Marneffe et al., 2006) and extract the main verb of each segment. We use 6161 clauses for the classification task, omitting clauses with *have* or *be* as the main verb and those where no main verb could be identified due to parsing errors (*none*). Table 1 shows inter-annotator agreement; Table 2 shows the confusion matrix for the two annotators. Our two annotators exhibit different preferences on the 598 cases where they disagree between DYNAMIC and STATIVE. Such differences in annotation prefer-

²Corpus freely available from www.coli.uni-saarland.de/~afried.

	DYNAMIC	STATIVE	BOTH
DYNAMIC	1444	201	54
STATIVE	168	697	20
BOTH	44	31	8

Table 3: **Asp-Ambig**: confusion matrix for two annotators. Cohen’s κ is 0.6.

ences are not uncommon (Beigman Klebanov et al., 2008). We observe higher agreement in the jokes and news subcorpora than for letters; texts in the letters subcorpora are largely argumentative and thus have a different rhetorical style than the more straightforward narratives and reports found in jokes. Overall, we find substantial agreement.

The data for our experiments uses the label DYNAMIC or STATIVE whenever annotators agree, and BOTH whenever they disagree or when at least one annotator marked the clause as BOTH, assuming that both readings are possible in such cases. Because we don’t want to model the authors’ personal view of the theory, we refrain from applying an adjudication step and model the data as is.

Asp-Ambig: (Brown) In order to facilitate a first study on ambiguous verbs, we select 20 frequent verbs from the list of ‘mixed’ verbs (see section 3) and for each mark 138 sentences. Sentences are extracted randomly from the Brown corpus, such that the distribution of stative/dynamic usages is expected to be natural. We present entire sentences to the annotators who mark the aspectual class of the verb in question as highlighted in the sentence. The data is processed in the same way as Asp-MASC, discarding instances with parsing problems. This results in 2667 instances. κ is 0.6, the confusion matrix is shown in Table 3. Details are listed in Table 10.

4 Model and Features

For predicting the aspectual class of verbs in context (STATIVE, DYNAMIC, BOTH), we assume a supervised learning setting and explore features mined from a large background corpus, distributional features, and instance-based features. If not indicated otherwise, experiments use a Random Forest classifier (Breiman, 2001) trained with the implementation and standard parameter settings from Weka (Hall et al., 2009).

4.1 Linguistic indicator features (LingInd)

This set of corpus-based features is a reimplementation of the linguistic indicators of Siegel

FEATURE	EXAMPLE
frequency	-
present	<i>says</i>
past	<i>said</i>
future	<i>will say</i>
perfect	<i>had won</i>
progressive	<i>is winning</i>
negated	<i>not/never</i>
particle	<i>up/in/...</i>
no subject	-

FEATURE	EXAMPLE
continuous adverb	<i>continually</i> <i>endlessly</i>
evaluation adverb	<i>better</i> <i>horribly</i>
manner adverb	<i>furiously</i> <i>patiently</i>
temporal adverb	<i>again</i> <i>finally</i>
in-PP	<i>in an hour</i>
for-PP	<i>for an hour</i>

Table 4: **LingInd** feature set and examples for lexical items associated with each indicator.

FEATURE	VALUES
part-of-speech tag of the verb	VB, VBG, VBN, ...
tense	present, past, future
progressive	true/false
perfect	true/false
voice	active/passive
grammatical dependents	WordNet lexname/POS

Table 5: Instance-based (**Inst**) features

and McKeown (2000), who show that (some of) these features correlate with either stative or dynamic verb types. We parse the AFE and XIE sections of Gigaword (Graff and Cieri, 2003) with the Stanford dependency parser. For each verb type, we obtain a normalized count showing how often it occurs with each of the indicators in Table 4, resulting in one value per feature per verb. For example, for the verb *fill*, the value of the feature `temporal-adverb` is 0.0085, meaning that 0.85% of the occurrences of *fill* in the corpus are modified by one of the temporal adverbs on the list compiled by Siegel (1998b). Tense, progressive, perfect and voice are extracted using a set of rules following Loaiciga et al. (2014).³

4.2 Distributional Features (Dist)

We aim to leverage existing, possibly noisy sets of representative stative, dynamic or mixed verb types extracted from LCS (see section 3), making up for unseen verbs and noise by averaging over distributional similarities. Using an existing large distributional model (Thater et al., 2011) estimated over the set of Gigaword documents marked as stories, for each verb type, we build a syntactically informed vector representing the contexts in which the verb occurs. We compute three numeric feature values per verb type, which correspond to the average cosine similarities with the verb types in each of the three seed sets.

³We thank the authors for providing us their code.

FEATURES	ACCURACY (%)
Baseline (Lemma)	83.6
LingInd	83.8
Inst	70.8
Inst+Lemma	83.7
Dist	83.4
LingInd+Inst+Dist+Lemma	84.1

Table 6: **Experiment 1:** SEEN verbs, using **Asp-MASC**. Baseline memorizes most frequent class per verb type in training folds.

4.3 Instance-based features (Inst)

Table 5 shows our set of instance-based syntactic and semantic features. In contrast to the above described type-based features, these features do not rely on a background corpus, but are extracted from the clause being classified. Tense, progressive, perfect and voice are extracted from dependency parses as described above. For features encoding grammatical dependents, we focus on a subset of grammatical relations. The feature value is either the WordNet lexical filename (e.g. *noun.person*) of the given relation’s argument or its POS tag, if the former is not available. We simply use the most frequent sense for the dependent’s lemma. We also include features that indicate, if there are any, the particle of the verb and its prepositional dependents. For the sentence *A little girl had just finished her first week of school*, the instance-based feature values would include *tense:past*, *subj:noun.person*, *dobj:noun.time* or *particle:none*.

5 Experiments

The experiments presented in this section aim to evaluate the effectiveness of the feature sets described in the previous section, focusing on the challenging cases of verb types unseen in the training data and highly ambiguous verbs. The feature **Lemma** indicates that the verb’s lemma is used as an additional feature.

Experiment 1: SEEN verbs

The setting of our first experiment follows Siegel and McKeown (2000). Table 6 reports results for 10-fold cross-validation, with occurrences of all verbs distributed evenly over the folds. No feature combination significantly⁴ outperforms the baseline of simply memorizing the most frequent class

⁴According to McNemar’s test with Yates’ correction for continuity, $p < 0.01$.

	FEATURES	ACCURACY (%)
1	Baseline	72.5
2	Dist	78.3*
3	LingInd	80.4*
4	LingInd+Dist	81.9*†

Table 7: **Experiment 2:** UNSEEN verb types, Logistic regression, **Asp-MASC**. Baseline labels everything with the most frequent class in the training set (DYNAMIC). *Significantly⁴ different from line 1. †Significantly⁴ different from line 3.

DATA	FEATURES	ACC. (%)
one-label verbs (1966 inst.)	Baseline	92.8
	LingInd	92.8
	Dist	92.6
	Inst+Lemma	91.4*
	LingInd+Inst+Lemma	92.4
multi-label verbs (4195 inst.)	Baseline	78.9
	LingInd	79.0
	Dist	79.0
	Inst	67.4*
	Inst+Lemma	79.9
	LingInd+Inst+Lemma	80.9*
	LingInd+Inst+Lemma+Dist	80.2*

Table 8: **Experiment 3:** ‘ONE- VS. MULTI-LABEL’ verbs, **Asp-MASC**. Baseline as in Table 6. *Indicates that result is significantly⁴ different from the respective baseline.

of a verb type in the respective training folds.

Experiment 2: UNSEEN verbs

This experiment shows a successful case of semi-supervised learning: while type-based feature values can be estimated from large corpora in an unsupervised way, some labeled training data is necessary to learn their best combination. This experiment specifically examines performance on verbs not seen in labeled training data. We use 10-fold cross validation but ensure that all occurrences of a verb type appear in the same fold: verb types in each test fold have *not* been seen in the respective training data, ruling out the Lemma feature. A Logistic regression classifier (Hall et al., 2009) works better here (using only numeric features), and we present results in Table 7. Both the LingInd and Dist features generalize across verb types, and their combination works best.

Experiment 3: ONE- vs. MULTI-LABEL verbs

For this experiment, we compute results separately for one-label verbs (those for which all instances in Asp-MASC have the same label) and

SYSTEM	CLASS	ACC.	P	R	F
baseline	micro-avg.	78.9	0.75	0.79	0.76
LingInd	DYNAMIC		0.84	0.95	0.89
+Inst	STATIVE		0.76	0.69	0.72
+Lemma	BOTH		0.51	0.24	0.33
	micro-avg.	80.9*	0.78	0.81	0.79

Table 9: **Experiment 3:** ‘MULTI-LABEL’, precision, recall and F-measure, detailed class statistics for the best-performing system from Table 8.

for multi-label verbs (instances have differing labels in Asp-MASC). We expect one-label verbs to have a strong predominant aspectual class, and multi-label verbs to be more flexible. Otherwise, the experimental setup is as in experiment 1. Results appear in Table 8. In each case, the linguistic indicator features again perform on par with the baseline. For multi-label verbs, the feature combination Lemma+LingInd+Inst leads to significant⁴ improvement of 2% gain in accuracy over the baseline; Table 9 reports detailed class statistics and reveals a gain in F-measure of 3 points over the baseline. To sum up, Inst features are essential for classifying multi-label verbs, and the LingInd features provide some useful prior. These results motivate the need for an instance-based approach.

Experiment 4: INSTANCE-BASED classification

For verbs with ambiguous aspectual class, type-based classification is not sufficient, as this approach selects a dominant sense for any given verb and then always assigns that. Therefore we propose handling ambiguous verbs separately. As Asp-MASC contains only few instances of each of the ambiguous verbs, we turn to the Asp-Ambig dataset. We perform a Leave-One-Out (LOO) cross validation evaluation, with results reported in Table 10.⁵ Using the Inst features alone (not shown in Table 10) results in a micro-average accuracy of only 58.1%: these features are only useful when combined with the feature Lemma. For classifying verbs whose most frequent class occurs less than 56% of the time, Lemma+Inst features are essential. Whether or not performance is improved by adding LingInd/Dist features, with their bias towards one aspectual class, depends on the verb type. It is an open research question which verb types should be treated in which way.

⁵ The third column also shows the outcome of using either only the Lemma, only LingInd or only Dist in LOO; all have almost the same outcome as using the majority class, numbers differ only after the decimal point.

VERB	# OF INST.	MAJORITY CLASS ⁵	Inst +Lemma	Inst +Lemma +LingInd +Dist
<i>feel</i>	128	96.1 STAT	93.0	93.8
<i>say</i>	138	94.9 DYN	93.5	93.5
<i>make</i>	136	91.9 DYN	91.9	91.2
<i>come</i>	133	88.0 DYN	87.2	87.2
<i>take</i>	137	85.4 DYN	85.4	85.4
<i>meet</i>	130	83.9 DYN	86.2	87.7
<i>stand</i>	130	80.0 STAT	79.2	83.1
<i>find</i>	137	74.5 DYN	69.3	68.8
<i>accept</i>	134	70.9 DYN	64.9	65.7
<i>hold</i>	134	56.0 BOTH	43.3	49.3
<i>carry</i>	136	55.9 DYN	55.9	58.1
<i>look</i>	138	55.8 DYN	72.5	74.6
<i>show</i>	133	54.9 DYN	69.2	68.4
<i>appear</i>	136	52.2 STAT	64.7	61.0
<i>follow</i>	122	51.6 BOTH	69.7	65.6
<i>consider</i>	138	50.7 DYN	61.6	70.3
<i>cover</i>	123	50.4 STAT	46.3	54.5
<i>fill</i>	134	47.8 DYN	66.4	62.7
<i>bear</i>	135	47.4 DYN	70.4	67.4
<i>allow</i>	135	37.8 DYN	48.9	51.9
micro-avg.	2667	66.3	71.0*	72.0*

Table 10: **Experiment 4:** INSTANCE-BASED. **Accuracy** (in %) on **Asp-Ambig**. *Differs significantly⁴ from the majority class baseline.

6 Discussion and conclusions

We have described a new, context-aware approach to automatically predicting aspectual class, including a new set of distributional features. We have also introduced two new data sets of clauses labeled for aspectual class. Our experiments show that in any setting where labeled training data is available, improvement over the most frequent class baseline can only be reached by integrating instance-based features, though type-based features (LingInd, Dist) may provide useful priors for some verbs and successfully predict predominant aspectual class for unseen verb types. In order to arrive at a globally well-performing system, we envision a multi-stage approach, treating verbs differently according to whether training data is available and whether or not the verb’s aspectual class distribution is highly skewed.

Acknowledgments We thank the anonymous reviewers, Omri Abend, Mike Lewis, Manfred Pinkal, Mark Steedman, Stefan Thater and Bonnie Webber for helpful comments, and our annotators A. Kirkland and R. Kühn. This research was supported in part by the MMCI Cluster of Excellence, and the first author is supported by an IBM PhD Fellowship.

References

- Emmon Bach. 1986. The algebra of events. *Linguistics and philosophy*, 9(1):5–16.
- Beata Beigman Klebanov, Eyal Beigman, and Daniel Diermeier. 2008. Analyzing disagreements. In *Proceedings of the Workshop on Human Judgements in Computational Linguistics*, pages 2–7. Association for Computational Linguistics.
- Steven Bethard. 2013. ClearTK-TimeML: A minimalist approach to TempEval 2013. In *Second Joint Conference on Lexical and Computational Semantics (*SEM)*, volume 2, pages 10–14.
- Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.
- Michael R. Brent. 1991. Automatic semantic classification of verbs from their syntactic contexts: an implemented classifier for stativity. In *Proceedings of the fifth conference on European chapter of the Association for Computational Linguistics*, pages 222–226. Association for Computational Linguistics.
- Nathanael Chambers. 2013. Navytime: Event and time ordering from raw text. In *Second Joint Conference on Lexical and Computational Semantics (*SEM)*, volume 2, pages 73–77.
- Francisco Costa and António Branco. 2012. Aspectual type and temporal relation classification. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 266–275. Association for Computational Linguistics.
- Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.
- Bonnie J. Dorr and Mari Broman Olsen. 1997. Deriving verbal and compositional lexical aspect for NLP applications. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 151–158. Association for Computational Linguistics.
- Bonnie J. Dorr. 1992. A two-level knowledge representation for machine translation: lexical semantics and tense/aspect. In *Lexical Semantics and Knowledge Representation*, pages 269–287. Springer.
- Bonnie J. Dorr. 2001. LCS verb database. Online software database of Lexical Conceptual Structures, University of Maryland, College Park, MD.
- David Dowty. 1979. *Word Meaning and Montague Grammar*. Reidel, Dordrecht.
- David Graff and Christopher Cieri. 2003. English gigaword.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.
- Nancy Ide, Christiane Fellbaum, Collin Baker, and Rebecca Passonneau. 2010. The manually annotated sub-corpus: a community resource for and by the people. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 68–73. Association for Computational Linguistics.
- Hyuckchul Jung and Amanda Stent. 2013. ATT1: Temporal annotation using big windows and rich syntactic and semantic features. In *Second Joint Conference on Lexical and Computational Semantics (*SEM)*, volume 2, pages 20–24.
- Graham Katz. 2003. On the stativity of the english perfect. *Perfect explorations*, pages 205–234.
- Judith L. Klavans and Martin Chodorow. 1992. Degrees of stativity: the lexical representation of verb aspect. In *Proceedings of the 14th conference on Computational Linguistics*, pages 1126–1131. Association for Computational Linguistics.
- Sharid Loaiciga, Thomas Meyer, and Andrei Popescu-Belis. 2014. English-French Verb Phrase Alignment in Europarl for Tense Translation Modeling. In *Language Resources and Evaluation Conference (LREC), Reykjavik, Iceland*.
- Michael C. McCord. 1990. *Slot Grammar*. Springer.
- Marc Moens and Mark J. Steedman. 1988. Temporal ontology and temporal reference. *Computational Linguistics*, 14(2):15–28.
- Alexander P.D. Mourelatos. 1978. Events, processes, and states. *Linguistics and philosophy*, 2(3):415–434.
- Alexander Nakhimovsky. 1988. Aspect, aspectual class, and the temporal structure of narrative. *Computational Linguistics*, 14(2):29–43.
- Rebecca Passonneau. 1988. A computational model of the semantics of tense and aspect. *Computational Linguistics*, Spring 1988.
- James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. 2003. The timebank corpus. In *Corpus linguistics*, volume 2003, page 40.
- Eric V. Siegel and Kathleen R. McKeown. 2000. Learning methods to combine linguistic indicators: Improving aspectual classification and revealing linguistic insights. *Computational Linguistics*, 26(4):595–628.

- Eric V. Siegel. 1998a. Disambiguating verbs with the WordNet category of the direct object. In *Proceedings of Workshop on Usage of WordNet in Natural Language Processing Systems*, Universite de Montreal.
- Eric V. Siegel. 1998b. *Linguistic Indicators for Language Understanding: Using machine learning methods to combine corpus-based indicators for aspectual classification of clauses*. Ph.D. thesis, Columbia University.
- Carlota S. Smith. 1991. *The Parameter of Aspect*. Kluwer, Dordrecht.
- Radu Soricut and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 149–156. Association for Computational Linguistics.
- Stefan Thater, Hagen Fürstenu, and Manfred Pinkal. 2011. Word meaning in context: A simple and effective vector model. In *IJCNLP*, pages 1134–1143.
- Naushad UzZaman, Hector Llorens, Leon Derczynski, Marc Verhagen, James Allen, and James Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second joint conference on lexical and computational semantics (*SEM)*, volume 2, pages 1–9.
- Zeno Vendler, 1957. *Linguistics in Philosophy*, chapter Verbs and Times, pages 97–121. Cornell University Press, Ithaca, New York.
- Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 75–80. Association for Computational Linguistics.
- Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. 2010. SemEval-2010 task 13: TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 57–62. Association for Computational Linguistics.

Combining Word Patterns and Discourse Markers for Paradigmatic Relation Classification

Michael Roth

ILCC, School of Informatics
University of Edinburgh
mroth@inf.ed.ac.uk

Sabine Schulte im Walde

Institut für Maschinelle Sprachverarbeitung
Universität Stuttgart
schulte@ims.uni-stuttgart.de

Abstract

Distinguishing between paradigmatic relations such as synonymy, antonymy and hypernymy is an important prerequisite in a range of NLP applications. In this paper, we explore discourse relations as an alternative set of features to lexico-syntactic patterns. We demonstrate that statistics over discourse relations, collected via explicit discourse markers as proxies, can be utilized as salient indicators for paradigmatic relations in multiple languages, outperforming patterns in terms of recall and F₁-score. In addition, we observe that markers and patterns provide complementary information, leading to significant classification improvements when applied in combination.

1 Introduction

Paradigmatic relations (such as synonymy, antonymy and hypernymy; cf. Murphy, 2003) are notoriously difficult to distinguish automatically, as first-order co-occurrences of the related words tend to be very similar across the relations. For example, in *The boy/girl/person loves/hates the cat*, the nominal co-hyponyms *boy*, *girl* and their hypernym *person* as well as the verbal antonyms *love* and *hate* occur in identical contexts, respectively. Vector space models, which represent words by frequencies of co-occurring words to enable comparisons in terms of *distributional similarity* (Schütze, 1992; Turney and Pantel, 2010), hence perform below their potential when inferring the type of relation that holds between two words. This distinction is crucial, however, in a range of tasks: in sentiment analysis, for example, words of the same and opposing polarity need to be distinguished; in textual entailment, systems further need to identify hypernymy because of directional inference requirements.

Accordingly, while there is a rich tradition on identifying word pairs of a single paradigmatic relation, there is little work that has addressed the distinction between two or more paradigmatic relations (cf. Section 2 for details). In more general terms, previous approaches to distinguishing between several semantic relations have predominantly relied on manually created knowledge sources, or lexico-syntactic patterns that can be automatically extracted from text. Each option comes with its own shortcomings: knowledge bases, on the one hand, are typically developed for a single language or domain, meaning that they might not generalize well; word patterns, on the other hand, are noisy and can be sparse for infrequent word pairs.

In this paper, we propose to strike a balance between availability and restrictedness by making use of *discourse markers*. This approach has several advantages: markers are frequently found across genres (Webber, 2009), they exist in many languages (Jucker and Yiv, 1998), and capture various semantic properties (Hutchinson, 2004). We implement discourse markers within a vector space model that aims to distinguish between the three paradigmatic relations *synonymy*, *antonymy* and *hypernymy* in German and in English, across the three word classes of nouns, verbs, adjectives. We examine the performance of discourse markers as vector space dimensions in isolation and also explore their contribution in combination with lexical patterns.

2 Related Work

As mentioned above, there is a rich tradition of research on identifying a single paradigmatic relations. Work on *synonyms* includes Edmonds and Hirst (2002), who employed a co-occurrence network and second-order co-occurrence, and Curran (2003), who explored word-based and syntax-based co-occurrence for thesaurus construction.

Van der Plas and Tiedemann (2006) compared a standard distributional approach against cross-lingual alignment; Erk and Padó (2008) defined a vector space model to identify synonyms and the substitutability of verbs. Most computational work on *hypernyms* was performed for nouns, cf. the lexico-syntactic patterns by Hearst (1992) and an extension of the patterns by dependency paths (Snow et al., 2004). Weeds et al. (2004), Lenci and Benotto (2012) and Santus et al. (2014) identified hypernyms in distributional spaces. Computational work on *antonyms* includes approaches that tested the co-occurrence hypothesis (Charles and Miller, 1989; Fellbaum, 1995), and approaches driven by text understanding efforts and contradiction frameworks (Harabagiu et al., 2006; Mohammad et al., 2008; de Marneffe et al., 2008).

Among the few approaches that distinguished *between* paradigmatic semantic relations, Lin et al. (2003) used patterns and bilingual dictionaries to retrieve distributionally similar words, and relied on clear antonym patterns such as ‘either X or Y’ in a post-processing step to distinguish synonyms from antonyms. The study by Mohammad et al. (2013) on the identification and ranking of opposites also included synonym/antonym distinction. Yih et al. (2012) developed an LSA approach incorporating a thesaurus, to distinguish the same two relations. Chang et al. (2013) extended this approach to induce vector representations that can capture multiple relations. Whereas the above mentioned approaches rely on additional knowledge sources, Turney (2006) developed a corpus-based approach to model relational similarity, addressing (among other tasks) the distinction between synonyms and antonyms. More recently, Schulte im Walde and Köper (2013) proposed to distinguish between the three relations antonymy, synonymy and hyponymy based on automatically acquired word patterns.

Regarding pattern-based approaches to identify and distinguish lexical semantic relations in more general terms, Hearst (1992) was the first to propose lexico-syntactic patterns as empirical pointers towards relation instances, focusing on hyponymy. Girju et al. (2003) applied a single pattern to distinguish pairs of nouns that are in a causal relationship from those that are not, and Girju et al. (2006) extended the work towards part-whole relations, applying a supervised, knowledge-intensive approach. Chklovski and Pantel (2004) were the first to apply pattern-

based relation extraction to verbs, distinguishing five non-disjoint relations (*similarity, strength, antonymy, enablement, happens-before*). Pantel and Pennacchiotti (2006) developed *Espresso*, a weakly-supervised system that exploits patterns in large-scale web data to distinguish between five noun-noun relations (*hypernymy, meronymy, succession, reaction, production*). Similarly to Girju et al. (2006), they used generic patterns, but relied on a bootstrapping cycle combined with reliability measures, rather than manual resources. Whereas each of the aforementioned approaches considers only one word class and clearly disjoint categories, we distinguish between paradigmatic relations that can be distributionally very similar and propose a unified framework for nouns, verbs and adjectives.

3 Baseline Model and Data Set

The task addressed in this work is to distinguish between synonymy, antonymy and hypernymy. As a starting point, we build on the approach and data set used by Schulte im Walde and Köper (2013, henceforth just S&K). In their work, frequency statistics over automatically acquired co-occurrence patterns were found to be good indicators for the paradigmatic relation that holds between two given words of the same word class. They further experimented with refinements of the vector space model, for example, by only considering patterns of a specific length, weighting by pointwise mutual information and applying thresholds based on frequency and reliability.

Baseline Model. We re-implemented the best model from S&K with the same setup: word pairs are represented by vectors, with each entry corresponding to one out of almost 100,000 patterns of lemmatized word forms (e.g., *X affect how you Y*). Each value is calculated as the log frequency of the corresponding pattern occurring between the word pairs in a corpus, based on exact match. For English, we use the ukWaC corpus (Baroni et al., 2009); for German, we rely on the COW corpus instead of deWaC, as it is larger and better balanced (Schäfer and Bildhauer, 2012).

Data Set. The evaluation data set by S&K is a collection of target and response words in German that has been collected via Amazon Mechanical Turk. The data contains a balanced amount of instances across word categories and relations, also taking into account corpus frequency, degree of ambiguity and semantic classes. In total, the

	S&K			Reimplemented		
	P	R	F ₁	P	R	F ₁
Nouns						
SYN-ANT	77.4	65.0	70.7	76.7	62.2	68.7
SYN-HYP	75.0	57.0	64.8	73.3	59.5	65.7
Verbs						
SYN-ANT	70.6	40.0	51.1	84.6	36.7	51.2
SYN-HYP	42.0	26.7	32.6	52.6	33.3	40.8
Adjectives						
SYN-ANT	88.9	66.7	76.2	94.1	66.7	78.0
SYN-HYP	68.4	54.2	60.5	65.0	54.2	59.1

Table 1: 2-way classification results by Schulte im Walde and Köper (2013) and our re-implementation. All numbers in percent.

data set consists of 692 pairs of instances, distributed over three word classes (nouns, verbs, adjectives) and three paradigmatic relations (synonymy, antonymy, hypernymy).

Intermediate Evaluation. We compare our re-implementation to the model by S&K using their 80% training and 20% test split, focusing on 2-way classifications involving synonymy. The results, summarized in Table 1, confirm that our re-implementation achieves similar results. Observed differences are probably an effect of the distinct corpora applied to induce patterns and counts.

We notice that the performance of both models strongly depends on the affected pair of relations and word category. For example, precision varies in the 2-way classification between synonymy and antonymy from 70.6% to 94.1%. Given the small amount of test data, some of the 80/20 splits might be better suited for the model than others. To avoid resulting bias effects, we perform our final evaluation using 5-fold cross-validation on a merged set of all training and test instances. To illustrate the performance of models in multiple languages, we further conduct experiments on a data set for English relation pairs that has been collected by Giulia Benotto and Alessandro Lenci, following the same methodology as the German collection. The English data set consists of 648 pairs of instances, also distributed over nouns, verbs, adjectives, and covering synonymy, antonymy, hypernymy.

4 Markers for Relation Classification

The aim of this work is to establish corpus statistics over discourse relations as a salient source of

CONTRAST	but, although, rather ...
RESTATEMENT	indeed, specifically, ...
INSTANTIATION	(for) example, instance, ...

Table 2: Examples of discourse relations/markers.

information to distinguish between paradigmatic relations. Our approach is motivated by linguistic studies that indicated a connection between discourse relations and lexical relations of words occurring in the respective discourse segments: Murphy et al. (2009) have shown, for example, that antonyms frequently serve as indicators for contrast relations in English and Swedish. More generally, pairs of word tokens have been identified as strong features for classifying discourse relations when no explicit discourse markers are available (Pitler et al., 2009; Biran and McKeown, 2013).

Whereas word pairs have frequently been used as features for disambiguating discourse relations, to the best of our knowledge, our approach is novel in that we are the first to apply discourse relations as features for classifying lexical relations. One reason for this might be that discourse relations in general are only available in manually annotated corpora. Previous work has shown, however, that such relations can be classified reliably given the presence of explicit discourse markers.¹ We hence rely on such markers as proxies for discourse relations (for examples, cf. Table 2).

4.1 Model and Hypothesis

We propose a vector space model that represents pairs of words using as features the discourse markers that occur between them. The underlying hypothesis of this model is as follows: if two phrases frequently co-occur with a specific discourse marker, then the discourse relation expressed by the corresponding marker should also indicate the relation between the words in the affected phrases. Following this hypothesis, contrast relations might indicate antonymy, whereas elaborations may indicate synonymy or hyponymy. Although such relations will not hold between every pair of words in two connected discourse segments, we hypothesize that correct instances (of all considered word classes) can be identified based on high relative frequency.

In our model, frequency statistics are computed over sentence-internal co-occurrences of

¹Pitler et al. (2008) report an accuracy of up to 93%.

word pairs and discourse markers. Since discourse relations are typically directed, we take into consideration whether a word occurs to the left or to the right of the respective marker. Accordingly, the features of our model are special cases of single-word patterns with an arbitrary number of wild card tokens (e.g., the marker feature ‘though’ corresponds to the pattern “ $X * \textit{though} * Y$ ”). Yet, our specific choice of features has several advantages: Whereas strict and potentially long patterns can be rare in text, discourse markers such as “however”, “for example” and “additionally” are frequently found across genres (Webber, 2009). Although combinations of tokens could also be replaced by wild cards in any automatically acquired pattern, this would generally lead to an exponentially growing feature space. In contrast, the set of discourse markers in our work is fixed: for English, we use 61 markers annotated in the Penn Discourse TreeBank 2.0 (Prasad et al., 2008); for German, we use 155 one-word translations of the English markers, as obtained from an online dictionary.^{2,3} Taking directionality into account, our vector space model consists of 2×61 and 2×155 features, respectively.

4.2 Development Set and Hyperparameters

We select the hyperparameters of our model using an independent development set, which we extract from the lexical resource GermaNet (Hamp and Feldweg, 1997). For each considered word category, we extract instances of synonymy, antonymy and hypernymy. In total, 1502 instances are identified, with 64 of them overlapping with the evaluation data set described in Section 3. Note though that the development set is not used for evaluation but only to select the following hyperparameters.

We experimented with different vector values (absolute frequency, log frequency, pointwise mutual information (PMI)), distance measures (cosine, euclidean) and normalization schemes. In contrast to S&K, who did not observe any improvements using PMI, we found it to perform best, combined with euclidean distance and no additional normalization. This finding might be an immediate effect of discourse markers being

²<http://dict.leo.org>

³We also experimented with larger sets of markers, including conjunctions and adverbials in sentence-initial positions, but did not notice any considerable effect. Future work could use manual sets of markers, e.g. those by Pasch et al. (2003), though such sets are only available in few languages.

generally more frequent than strict word patterns, which also leads to more reliable PMI values.

5 Evaluation

In our evaluation, we assess the performance of the marker-based model and demonstrate the benefits of incorporating discourse markers into a pattern-based model, which we apply as a baseline. We evaluate on several data sets: the collection of target-response pairs in German from previous work, and a similar data set that was collected for English target words (cf. Section 3); for comparison reasons, we also apply our models to the balanced data set of related and unrelated noun pairs by Yap and Baldwin (2009).⁴ We perform 3-way and 2-way relation classification experiments, using 5-fold cross-validation and a nearest centroid classifier (as applied by S&K).

Results. The 3-way classification results of the baseline and our marker-based model are summarized in Table 3, with best results for each setting marked in bold. On the German data set, our model always outperforms a random baseline (33% F_1 -score). The results on the English data set are overall a bit lower, possibly due to corpus size. In almost all classification tasks, our marker-based model achieves a higher recall and F_1 -score than the pattern-based approach. The precision results of the marker-based model are overall below the pattern-based model. This drop in performance does not come as a surprise though, considering that the model only makes use of 122 and 310 features, in comparison to tens of thousands of features in the pattern approach.

A randomized significance test over classified instances (cf. Yeh, 2000) revealed that only two differences in results are significant. We hypothesize that one reason for this outcome might be that both models cover complementary sets of instances. To verify this hypothesis, we apply a combined model, which is based on a weighted linear combination of distances computed by the two individual models.⁵ As displayed in Table 3, this combined model yields further improvements

⁴Note that we could, in principle, also apply our models to unbalanced data. Our main focus lies however on examining the direct impact of different feature sets. We hence decided to keep the evaluation setup simple and used a classifier that does not take into account class frequency.

⁵We determined the best weights on the development set and found these to be 0.9 and 0.1 for the output of the pattern-based and marker-based model, respectively.

		Nouns			Verbs			Adjectives		
		P	R	F ₁	P	R	F ₁	P	R	F ₁
German	Patterns	55.6	40.8	47.0	55.6	35.6	43.4	53.5	41.1	46.5
	Markers	42.6	38.7	40.5	48.4	46.2**	47.3	51.1	48.6	49.9
	Combined	50.4	45.7*	48.0	52.6	50.2**	51.4**	53.4	50.8**	52.1
English	Patterns	46.4	28.0	34.9	44.7	28.5	34.8	56.6	32.1	41.0
	Markers	39.0	34.3	36.5	38.3	36.3	37.2	50.0	41.2**	45.2
	Combined	43.0	37.8**	40.3*	41.8	39.6**	40.7*	53.5	44.4**	48.5**

Table 3: 3-way classification results using 5-fold cross-validation. All numbers in percent. Asterisks indicate significant differences to the pattern-based baseline model (* p<0.10, ** p<0.05).

Combined model	German			English		
	P	R	F ₁	P	R	F ₁
	Nouns					
SYN-ANT	61.7	55.7	58.5	52.9	44.2	48.2
SYN-HYP	66.5	60.4	63.3	62.2	58.6	60.4
ANT-HYP	70.9	64.6	67.6	59.1	50.6	54.5
	Verbs					
SYN-ANT	58.9	55.0	56.8	49.6	45.8	47.6
SYN-HYP	67.6	64.0	65.8	66.4	63.0	64.7
ANT-HYP	67.3	66.4	66.9	62.9	60.7	61.8
	Adjectives					
SYN-ANT	74.8	69.4	72.0	67.0	56.6	61.3
SYN-HYP	58.0	56.1	57.0	56.4	46.0	50.7
ANT-HYP	73.7	70.7	72.2	69.8	57.8	63.2

Table 4: 2-way results of the combined model. Bold numbers indicate improvements over both individual models. All numbers in percent.

in recall and F₁-score, leading to the best 3-way classification results. All gains in recall are significant, confirming that the single models indeed contribute complementary information. For example, only the pattern-based model classifies “intentional”–“accidental” as antonyms, and only the marker-based model predicts the correct relation for “double”–“multiple” (hypernymy). The combined model classifies both pairs correctly.

Table 4 further assesses the strength of the combined model on the 2-way classifications. The table highlights results indicating improvements over *both* individual models. We observe that the combined model achieves the best recall and F₁-score in 15 out of 18 cases.

Relation	SYN	ANT	HYP
Patterns	0.97	0.97	0.94
Markers	0.77*	0.82*	0.91*
Combined	0.93*	0.98	0.96*

Table 5: Results in F₁-score on the balanced data set by Yap and Baldwin (* p<0.05).

A final experiment is performed on the data set by Yap and Baldwin (2009) to see whether our models can also distinguish word pairs of individual relations from unrelated pairs of words. The results, listed in Table 5, show that the marker-based model cannot perform this task as well as the pattern-based model. The combined model, however, outperforms both individual models in 2 out of 3 cases. Despite their simplicity, our models achieve results close to the F₁-scores reported by Yap and Baldwin (0.98–0.99), who employed syntactic pre-processing and an SVM-based classifier, and experimented with different corpora.

6 Conclusions

In this paper, we proposed to use discourse markers as indicators for paradigmatic relations between words and demonstrated that a small set of such markers can achieve higher recall than a pattern-based model with tens of thousands of features. Combining patterns and markers can further improve results, leading to significant gains in recall and F₁. As our new model only relies on a raw corpus and a fixed list of discourse markers, it can easily be extended to other languages.

Acknowledgments

The research presented in this paper was funded by the DFG grant SCHU-2580/2-1 and the DFG Heiselberg Fellowship SCHU-2580/1-1.

References

- Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226.
- Or Biran and Kathleen McKeown. 2013. Aggregated word pair features for implicit discourse relation disambiguation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 69–73, Sofia, Bulgaria, August.
- Kai-Wei Chang, Wen-tau Yih, and Christopher Meek. 2013. Multi-relational latent semantic analysis. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1602–1612, Seattle, Washington, USA, October.
- Walter G. Charles and George A. Miller. 1989. Contexts of antonymous adjectives. *Applied Psycholinguistics*, 10(3):357–375.
- Tim Chklovski and Patrick Pantel. 2004. VerbOcean: Mining the Web for fine-grained semantic verb relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, 25–26 July 2004, pages 33–40.
- James Curran. 2003. *From Distributional to Semantic Similarity*. Ph.D. thesis, Institute for Communication and Collaborative Systems, School of Informatics, University of Edinburgh.
- Marie-Catherine de Marneffe, Anna N. Rafferty, and Christopher D. Manning. 2008. Finding contradictions in text. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1039–1047, Columbus, Ohio, USA.
- Philip Edmonds and Graeme Hirst. 2002. Near-synonymy and lexical choice. *Computational Linguistics*, 28(2):105–144.
- Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, Waikiki, Honolulu, Hawaii, 25-27 October 2008.
- Christiane Fellbaum. 1995. Co-occurrence and antonymy. *International Journal of Lexicography*, 8(4):281–303.
- Roxana Girju, Adriana Badulescu, and Dan Moldovan. 2003. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, Edmonton, Alberta, Canada, 27 May –1 June 2003, pages 80–87.
- Roxana Girju, Adriana Badulescu, and Dan Moldovan. 2006. Automatic discovery of part-whole relations. *Computational Linguistics*, 32(1):83–135.
- Birgit Hamp and Helmut Feldweg. 1997. GermaNet - a lexical-semantic net for German. In *Proceedings of the Workshop on Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications at ACL/EACL-97*, Madrid, Spain, 12 July 1997, pages 9–15.
- Sanda Harabagiu, Andrew Hickl, and Finley Lacatusu. 2006. Negation, contrast and contradiction in text processing. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 755–762.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 15th International Conference on Computational Linguistics*, Nantes, France, 23-28 August 1992, pages 539–545.
- Ben Hutchinson. 2004. Acquiring the meaning of discourse markers. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, 21–26 July 2004, pages 685–692.
- Andreas H. Jucker and Zael Yiv, editors. 1998. *Discourse Markers: Descriptions and Theory*, volume 57 of *Discourse & Beyond New Series*. John Benjamin Publishing Company.
- Alessandro Lenci and Giulia Benotto. 2012. Identifying hypernyms in distributional semantic spaces. In *Proceedings of the First Joint Conference on Lexical and Computational Semantic*, pages 75–79.
- Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. 2003. Identifying synonyms among distributionally similar words. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 1492–1493. Morgan Kaufmann Publishers Inc.
- Saif M. Mohammad, Bonnie Dorr, and Graeme Hirst. 2008. Computing word-pair antonymy. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 982–991, Honolulu, Hawaii, USA.
- Saif M. Mohammad, Bonnie J. Dorr, Graeme Hirst, and Peter D. Turney. 2013. Computing lexical contrast. *Computational Linguistics*, 39(3):555–590.
- M. Lynne Murphy, Carita Paradis, Caroline Willners, and Steven Jones. 2009. Discourse functions of antonymy: A cross-linguistic investigation of Swedish and English. *Journal of Pragmatics*, 41(11):2159–2184.
- M. Lynne Murphy. 2003. *Semantic relations and the lexicon*. Cambridge University Press.

- Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia, 17–21 July 2006, pages 113–120.
- Renate Pasch, Ursula Brausse, Eva Breindl, and Ulrich Wassner. 2003. *Handbuch der deutschen Konnektoren*. Walter de Gruyter, Berlin.
- Emily Pitler and Ani Nenkova. 2008. Revisiting readability: A unified framework for predicting text quality. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 186–195, Honolulu, Hawaii, October.
- Emily Pitler, Annie Louis, and Ani Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. 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 683–691, Suntec, Singapore, August.
- Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Milt-sakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morocco, May.
- Enrico Santus, Alessandro Lenci, Qin Lu, and Sabine Schulte im Walde. 2014. Chasing hypernyms in vector spaces with entropy. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 38–42, Gothenburg, Sweden.
- Roland Schäfer and Felix Bildhauer. 2012. Building large corpora from the web using a new efficient tool chain. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 486–493, Istanbul, Turkey, May.
- Sabine Schulte im Walde and Maximilian Köper. 2013. Pattern-based distinction of paradigmatic relations for German nouns, verbs, adjectives. In *Language Processing and Knowledge in the Web*, pages 184–198. Springer.
- Hinrich Schütze. 1992. Dimensions of meaning. In *In Proceedings of Supercomputing*, pages 787–796.
- Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. In *Advances in Neural Information Processing Systems*, volume 17, pages 1297–1304.
- Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.
- Peter D. Turney. 2006. Similarity of semantic relations. *Computational Linguistics*, 32(3):379–416.
- Lonneke Van der Plas and Jörg Tiedemann. 2006. Finding synonyms using automatic word alignment and measures of distributional similarity. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 866–873.
- Bonnie Webber. 2009. Genre distinctions for discourse in the Penn TreeBank. 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 674–682, Suntec, Singapore, August.
- Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 1015–1021.
- Willy Yap and Timothy Baldwin. 2009. Experiments on pattern-based relation learning. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pages 1657–1660.
- Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 947–953, Saarbrücken, Germany, August.
- Wen-tau Yih, Geoffrey Zweig, and John Platt. 2012. Polarity inducing latent semantic analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1212–1222, Jeju Island, Korea, July.

Applying a Naive Bayes Similarity Measure to Word Sense Disambiguation

Tong Wang
University of Toronto
tong@cs.toronto.edu

Graeme Hirst
University of Toronto
gh@cs.toronto.edu

Abstract

We replace the overlap mechanism of the Lesk algorithm with a simple, general-purpose Naive Bayes model that measures *many-to-many* association between two sets of random variables. Even with simple probability estimates such as maximum likelihood, the model gains significant improvement over the Lesk algorithm on word sense disambiguation tasks. With additional lexical knowledge from WordNet, performance is further improved to surpass the state-of-the-art results.

1 Introduction

To disambiguate a homonymous word in a given context, Lesk (1986) proposed a method that measured the degree of overlap between the glosses of the target and context words. Known as the Lesk algorithm, this simple and intuitive method has since been extensively cited and extended in the word sense disambiguation (WSD) community. Nonetheless, its performance in several WSD benchmarks is less than satisfactory (Kilgarriff and Rosenzweig, 2000; Vasilescu et al., 2004). Among the popular explanations is a key limitation of the algorithm, that “Lesk’s approach is very sensitive to the exact wording of definitions, so the absence of a certain word can radically change the results.” (Navigli, 2009).

Compounding this problem is the fact that many Lesk variants limited the concept of overlap to the literal interpretation of string matching (with their own variants such as length-sensitive matching (Banerjee and Pedersen, 2002), etc.), and it was not until recently that overlap started to take on other forms such as tree-matching (Chen et al., 2009) and vector space models (Abdalgader and Skabar, 2012; Raviv et al., 2012; Patwardhan and Pedersen, 2006). To address this limitation, a

Naive Bayes model (NBM) is proposed in this study as a novel, probabilistic treatment of overlap in gloss-based WSD.

2 Related Work

In the extraordinarily rich literature on WSD, we focus our review on those closest to the topic of Lesk and NBM. In particular, we opt for the “simplified Lesk” (Kilgarriff and Rosenzweig, 2000), where inventory senses are assessed by gloss-context overlap rather than gloss-gloss overlap. This particular variant prevents proliferation of gloss comparison on larger contexts (Mihalcea et al., 2004) and is shown to outperform the original Lesk algorithm (Vasilescu et al., 2004).

To the best of our knowledge, NBMs have been employed exclusively as classifiers in WSD — that is, in contrast to their use as a similarity measure in this study. Gale et al. (1992) used NB classifier resembling an information retrieval system: a WSD instance is regarded as a document d , and candidate senses are scored in terms of “relevance” to d . When evaluated on a WSD benchmark (Vasilescu et al., 2004), the algorithm compared favourably to Lesk variants (as expected for a supervised method). Pedersen (2000) proposed an ensemble model with multiple NB classifiers differing by context window size. Hristea (2009) trained an unsupervised NB classifier using the EM algorithm and empirically demonstrated the benefits of WordNet-assisted (Fellbaum, 1998) feature selection over local syntactic features.

Among Lesk variants, Banerjee and Pedersen (2002) extended the gloss of both inventory senses and the context words to include words in their related synsets in WordNet. Senses were scored by the sum of overlaps across all relation pairs, and the effect of individual relation pairs was evaluated in a later work (Banerjee and Pedersen, 2003). Overlap was assessed by string matching, with the number of matching words squared so as to assign

higher scores to multi-word overlaps.

Breaking away from string matching, Wilks et al. (1990) measured overlap as similarity between gloss- and context-vectors, which were aggregated word vectors encoding second order co-occurrence information in glosses. An extension by Patwardhan and Pedersen (2006) differentiated context word senses and extended shorter glosses with related glosses in WordNet. Patwardhan et al. (2003) measured overlap by *concept similarity* (Budanitsky and Hirst, 2006) between each inventory sense and the context words. Gloss overlaps from their earlier work actually out-performed all five similarity-based methods.

More recently, Chen et al. (2009) proposed a tree-matching algorithm that measured gloss-context overlap as the weighted sum of dependency-induced lexical distance. Abdalgader and Skabar (2012) constructed a *sentential* similarity measure (Li et al., 2006) using *lexical* similarity measures (Budanitsky and Hirst, 2006), and overlap was measured by the cosine of their respective sentential vectors. A related approach (Raviv et al., 2012) also used Wikipedia-induced concepts to encoded sentential vectors. These systems compared favourably to existing methods in WSD performance, although by using sense frequency information, they are essentially supervised methods.

Distributional methods have been used in many WSD systems in quite different flavours than the current study. Kilgarriff and Rosenzweig (2000) proposed a Lesk variant where each gloss word is weighted by its *idf* score in relation to all glosses, and gloss-context association was incremented by these weights rather than binary, overlap counts. Miller et al. (2012) used distributional thesauri as a knowledge base to increase overlaps, which were, again, assessed by string matching.

In conclusion, the majority of Lesk variants focused on extending the gloss to increase the chance of overlapping, while the proposed NBM aims to make better use of the limited lexical knowledge available. In contrast to string matching, the probabilistic nature of our model offers a “softer” measurement of gloss-context association, resulting in a novel approach to unsupervised WSD with state-of-the-art performance in more than one WSD benchmark (Section 4).

3 Model and Task Descriptions

3.1 The Naive Bayes Model

Formally, given two sets $\mathbf{e} = \{e_i\}$ and $\mathbf{f} = \{f_j\}$ each consisting of multiple random events, the proposed model measures the probabilistic association $p(\mathbf{f}|\mathbf{e})$ between \mathbf{e} and \mathbf{f} . Under the assumption of conditional independence among the events in each set, a Naive Bayes treatment of the measure can be formulated as:

$$p(\mathbf{f}|\mathbf{e}) = \prod_j p(f_j|\{e_i\}) = \prod_j \frac{p(\{e_i\}|f_j)p(f_j)}{p(\{e_i\})} \\ = \frac{\prod_j [p(f_j) \prod_i p(e_i|f_j)]}{\prod_j \prod_i p(e_i)}, \quad (1)$$

In the second expression, Bayes’s rule is applied not only to take advantage of the conditional independence among e_i ’s, but also to facilitate probability estimation, since $p(\{e_i\}|f_j)$ is easier to estimate in the context of WSD, where sample spaces of \mathbf{e} and \mathbf{f} become asymmetric (Section 3.2).

3.2 Model Application in WSD

In the context of WSD, \mathbf{e} can be regarded as an instance of a polysemous word w , while \mathbf{f} represents certain lexical knowledge about the sense s of w manifested by \mathbf{e} .¹ WSD is thus formulated as identifying the sense s^* in the sense inventory \mathcal{S} of w s.t.:

$$s^* = \arg \max_{s \in \mathcal{S}} p(\mathbf{f}|\mathbf{e}) \quad (2)$$

In one of their simplest forms, e_i ’s correspond to co-occurring words in the instance of w , and f_j ’s consist of the gloss words of sense s . Consequently, $p(\mathbf{f}|\mathbf{e})$ is essentially measuring the association between context words of w and definition texts of s , i.e., the gloss-context association in the simplified Lesk algorithm (Kilgarriff and Rosenzweig, 2000). A major difference, however, is that instead of using hard, overlap counts between the two sets of words from the gloss and the context, this probabilistic treatment can implicitly model the distributional similarity among the elements e_i and f_j (and consequently between the sets \mathbf{e} and \mathbf{f}) over a wider range of contexts. The result is a “softer” proxy of association than the binary view of overlaps in existing Lesk variants.

The foregoing discussion offers a second motivation for applying Bayes’s rule on the second

¹Think of the notations \mathbf{e} and \mathbf{f} mnemonically as *examples* and *features*, respectively.

Senses	Hypernyms	Hyponyms	Synonyms
<i>factory</i>	building complex, complex	brewery, factory, mill, ...	works, industrial plant
<i>life form</i>	organism, being	perennial, crop...	flora, plant life

Table 1: Lexical knowledge for the word *plant* under its two meanings *factory* and *life form*.

expression in Equation (1): it is easier to estimate $p(e_i|f_j)$ than $p(f_j|e_i)$, since the vocabulary for the lexical knowledge features (f_j) is usually more limited than that of the contexts (e_i) and hence estimation of the former suffices on a smaller amount of data than that of the latter.

3.3 Incorporating Additional Lexical Knowledge

The input of the proposed NBM is bags of words, and thus it is straightforward to incorporate various forms of lexical knowledge (LK) for word senses: by concatenating a tokenized knowledge source to the existing knowledge representation \mathbf{f} , while the similarity measure remains unchanged.

The availability of LK largely depends on the sense inventory used in a WSD task. WordNet senses are often used in Senseval and SemEval tasks, and hence senses (or synsets, and possibly their corresponding word forms) that are semantic related to the inventory senses under WordNet relations are easily obtainable and have been exploited by many existing studies.

As pointed out by Patwardhan et al. (2003), however, “not all of these relations are equally helpful.” Relation pairs involving hyponyms were shown to result in better F-measure when used in gloss overlaps (Banerjee and Pedersen, 2003). The authors attributed the phenomenon to the multitude of hyponyms compared to other relations. We further hypothesize that, beyond sheer numbers, synonyms and hyponyms offer stronger semantic specification that helps distinguish the senses of a given ambiguous word, and thus are more effective knowledge sources for WSD.

Take the word *plant* for example. Selected hypernyms, hyponyms, and synonyms pertaining to its two senses *factory* and *life form* are listed in Table 1. Hypernyms can be overly general terms (e.g., *being*). Although conceptually helpful for humans in coarse-grained WSD, this generality is

likely to inflate the hypernyms’ probabilistic estimation. Hyponyms, on the other hand, help specify their corresponding senses with information that is possibly missing from the often overly brief glosses: the many technical terms as hyponyms in Table 1 — though rare — are likely to occur in the (possibly domain-specific) contexts that are highly typical of the corresponding senses. Particularly for the NBM, the co-occurrence is likely to result in stronger gloss-definition associations when similar contexts appear in a WSD instance.

We also observe that some semantically related words appear under rare senses (e.g., *still* as an alcohol-manufacturing plant, and *annual* as a one-year-life-cycle plant; omitted from Table 1). This is a general phenomenon in gloss-based WSD and is beyond the scope of the current discussion.² Overall, all three sources of LK may complement each other in WSD tasks, with hyponyms particularly promising in both quantity and quality compared to hypernyms and synonyms.³

3.4 Probability Estimation

A most open-ended question is how to estimate the probabilities in Equation (1). In WSD in particular, the estimation concerns the marginal and conditional probabilities of and between word tokens. Many options are available to this end in statistical machine learning (MLE, MAP, etc.), information theory (Church and Hanks, 1990; Turney, 2001), as well as the rich body of research in lexical semantic similarity (Resnik, 1995; Jiang and Conrath, 1997; Budanitsky and Hirst, 2006).

Here we choose maximum likelihood — not only for its simplicity, but also to demonstrate model strength with a relatively crude probability estimation. To avoid underflow, Equation (1) is estimated as the following log probability:

$$\begin{aligned} & \sum_i \log \frac{c(f_j)}{c(\cdot)} + \sum_i \sum_j \log \frac{c(e_i, f_j)}{c(f_j)} - |\mathbf{f}| \sum_j \log \frac{c(e_i)}{c(\cdot)} \\ = & (1 - |\mathbf{e}|) \sum_i \log c(f_j) - |\mathbf{f}| \sum_j \log c(e_i) \\ & + \sum_i \sum_j \log c(e_i, f_j) + |\mathbf{f}|(|\mathbf{e}| - 1) \log c(\cdot), \end{aligned}$$

where $c(x)$ is the count of word x , $c(\cdot)$ is the corpus

²We do, however, refer curious readers to the work of Ravi et al. (2012) for a novel treatment of a similar problem.

³Note that LK expansion is a feature of our model rather than a requirement. What type of knowledge to include is eventually a decision made by the user based on the application and LK availability.

size, $c(x, y)$ is the joint count of x and y , and $|\mathbf{v}|$ is the dimension of vector \mathbf{v} .

Nonetheless, we do investigate how model performance responds to estimation quality. Specifically in WSD, a *source corpus* is defined as the source of the majority of the WSD instances in a given dataset, and a *baseline corpus* of a smaller size and less resemblance to the instances is used for all datasets. The assumption is that a source corpus offers better estimates for the model than the baseline corpus, and difference in model performance is expected when using probability estimation of different quality.

4 Evaluation

4.1 Data, Scoring, and Pre-processing

Various aspects of the model discussed in Section 3 are evaluated in the English lexical sample tasks from Senseval-2 (Edmonds and Cotton, 2001) and SemEval-2007 (Pradhan et al., 2007). Training sections are used as development data and test sections held out for final testing. Model performance is evaluated in terms of WSD accuracy using Equation (2) as the scoring function. Accuracy is defined as the number of correct responses over the number of instances. Because it is a rare event for the NBM to produce identical scores,⁴ the model always proposes a unique answer and accuracy is thus equivalent to F-score commonly used in existing reports.

Multiword expressions (MWEs) in the Senseval-2 sense inventory are not explicitly marked in the contexts. Several of the top-ranking systems implemented their own MWE detection algorithms (Kilgarriff and Rosenzweig, 2000; Litkowski, 2002). Without digressing to the details of MWE detection — and meanwhile, to ensure fair comparison with existing systems — we implement two variants of the prediction module, one completely ignorant of MWE and defaulting to INCORRECT for all MWE-related answers, while the other assuming perfect MWE detection and performing regular disambiguation algorithm on the MWE-related senses (*not* defaulting to CORRECT). All results reported for Senseval-2 below are harmonic means of the two outcomes.

Each inventory sense is represented by a set of LK tokens (e.g., definition texts, synonyms, etc.)

⁴This has never occurred in the hundreds of thousands of runs in our development process.

from their corresponding WordNet synset (or in the coarse-grained case, a concatenation of tokens from all synsets in a sense group). The *MIT-JWI* library (Finlayson, 2014) is used for accessing WordNet. Usage examples in glosses (included by the library by default) are removed in our experiments.⁵

Basic pre-processing is performed on the contexts and the glosses, including lower-casing, stopword removal, lemmatization on both datasets, and tokenization on the Senseval-2 instances.⁶ *Stanford CoreNLP*⁷ is used for lemmatization and tokenization. Identical procedures are applied to all corpora used for probability estimation.

Binomial test is used for significance testing, and with one exception explicitly noted in Section 4.3, all differences presented are statistically highly significant ($p < 0.001$).

4.2 Comparing Lexical Knowledge Sources

To study the effect of different types of LK in WSD (Section 3.3), for each inventory sense, we choose synonyms (*syn*), hypernyms (*hpr*), and hyponyms (*hpo*) as extended LK in addition to its gloss. The WSD model is evaluated with gloss-only (*glo*), individual extended LK sources, and the combination of all four sources (*all*). The results are listed in Table 2 together with existing results (1st and 2nd correspond to the results of the top two unsupervised methods in each dataset).⁸

By using only glosses, the proposed model already shows statistically significant improvement over the basic Lesk algorithm (92.4% and 140.5% relative improvement in Senseval-2 coarse- and fine-grained tracks, respectively).⁹ Moreover, comparison between coarse- and fine-grained tracks reveals interesting properties of different LK sources. Previous hypotheses (Section 3.3) are empirically confirmed that WSD perfor-

⁵We also compared the two Lesk baselines (with and without usage examples) on the development data but did not observe significant differences as reported by Kilgarriff and Rosenzweig (2000).

⁶The SemEval-2007 instances are already tokenized.

⁷<http://nlp.stanford.edu/software/corenlp.shtml>.

⁸We excluded the results of *UNED* (Fernández-Amorós et al., 2001) in Senseval-2 because, by using sense frequency information that is only obtainable from sense-annotated corpora, it is essentially a supervised system.

⁹Comparisons are made against the simplified Lesk algorithm (Kilgarriff and Rosenzweig, 2000) without usage examples. The comparison is unavailable in SemEval2007 since we have not found existing experiments with this exact configuration.

Dataset	<i>glo</i>	<i>syn</i>	<i>hpr</i>	<i>hpo</i>	<i>all</i>	1st	2nd	<i>Lesk</i>
<i>Senseval-2 Coarse</i>	.475	.478	.494	.518	.523	.469	.367	.262
<i>Senseval-2 Fine</i>	.362	.371	.326	.379	.388	.412	.293	.163
<i>SemEval-2007</i>	.494	.511	.507	.550	.573	.538	.521	–

Table 2: Lexical knowledge sources and WSD performance (*F-measure*) on the Senseval-2 (fine- and coarse-grained) and the SemEval-2007 dataset.

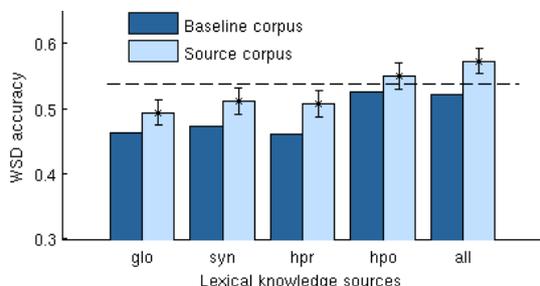


Figure 1: Model response to probability estimates of different quality on the SemEval-2007 dataset. Error bars indicate confidence intervals ($p < .001$), and the dashed line corresponds to the best reported result.

mance benefits most from hyponyms and least from hypernyms. Specifically, highly similar, fine-grained sense candidates apparently share more hypernyms in the fine-grained case than in the coarse-grained case; adding to the generality of hypernyms (both semantic and distributional), we postulate that their probability in the NBM is uniformly inflated among many sense candidates, and hence they decrease in distinguishability. Synonyms might help with regard to semantic specification, though their limited quantity also limits their benefits. These patterns on the LK types are consistent in all three experiments.

When including all four LK sources, our model outperforms the state-of-the-art systems with statistical significance in both coarse-grained tasks. For the fine-grained track, it achieves 2nd place after that of Tugwell and Kilgarriff (2001), which used a decision list (Yarowsky, 1995) on *manually selected* corpora evidence for each inventory sense, and thus is not subject to loss of distinguishability in the glosses as Lesk variants are.

4.3 Probability Estimation

To evaluate model response to probability estimation of different quality (Section 3.4), source corpora are chosen as the majority value of the *doc-source* attribute of instances in each dataset,

namely, the *British National Corpus* for Senseval-2 (94%) and the *Wall Street Journal* for SemEval-2007 (86%). The *Brown Corpus* is shared by both datasets as the baseline corpus. Figure 1 shows the comparison on the SemEval-2007 dataset. Across all experiments, higher WSD accuracy is consistently witnessed using the source corpus; differences are statistically highly significant except for *hpo* (which is significant with $p < 0.01$).

5 Conclusions and Future Work

We have proposed a general-purpose Naive Bayes model for measuring association between two sets of random events. The model replaced string matching in the Lesk algorithm for word sense disambiguation with a probabilistic measure of gloss-context overlap. The base model on average more than doubled the accuracy of Lesk in Senseval-2 on both fine- and coarse-grained tracks. With additional lexical knowledge, the model also outperformed state of the art results with statistical significance on two coarse-grained WSD tasks.

For future work, we plan to apply the model in other shared tasks, including open-text WSD, so as to compare with more recent Lesk variants. We would also like to explore how to incorporate syntactic features and employ alternative statistical methods (e.g., parametric models) to improve probability estimation and inference. Other NLP problems involving compositionality in general might also benefit from the proposed many-to-many similarity measure.

Acknowledgments

This study is funded by the Natural Sciences and Engineering Research Council of Canada. We thank Afsaneh Fazly, Navdeep Jaitly, and Varada Kolhatkar for the many inspiring discussions, as well as the anonymous reviewers for their constructive advice.

References

- Khaled Abdalgader and Andrew Skabar. Unsupervised similarity-based word sense disambiguation using context vectors and sentential word importance. *ACM Transactions on Speech and Language Processing*, 9(1):2:1–2:21, May 2012.
- Satanjeev Banerjee and Ted Pedersen. An adapted Lesk algorithm for word sense disambiguation using WordNet. In *Computational Linguistics and Intelligent Text Processing*, pages 136–145. Springer, 2002.
- Satanjeev Banerjee and Ted Pedersen. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, volume 3, pages 805–810, 2003.
- Alexander Budanitsky and Graeme Hirst. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47, 2006.
- Ping Chen, Wei Ding, Chris Bowes, and David Brown. A fully unsupervised word sense disambiguation method using dependency knowledge. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 28–36, Stroudsburg, PA, USA, 2009.
- Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29, 1990.
- Philip Edmonds and Scott Cotton. Senseval-2: Overview. In *Proceedings of the 2nd International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5. Association for Computational Linguistics, 2001.
- Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, 1998.
- David Fernández-Amorós, Julio Gonzalo, and Felisa Verdejo. The UNED systems at Senseval-2. In *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 75–78. Association for Computational Linguistics, 2001.
- Mark Alan Finlayson. Java libraries for accessing the Princeton WordNet: Comparison and evaluation. In *Proceedings of the 7th Global Wordnet Conference*, Tartu, Estonia, 2014.
- William Gale, Kenneth Church, and David Yarowsky. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26(5-6):415–439, 1992.
- Florentina Hristea. Recent advances concerning the usage of the Naïve Bayes model in unsupervised word sense disambiguation. *International Review on Computers & Software*, 4(1), 2009.
- Jay Jiang and David Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *Proceedings of International Conference on Research in Computational Linguistics*, 1997.
- Adam Kilgarriff and Joseph Rosenzweig. Framework and results for English Senseval. *Computers and the Humanities*, 34(1-2):15–48, 2000.
- Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, pages 24–26, New York, New York, USA, 1986.
- Yuhua Li, David McLean, Zuhair A Bandar, James D O’Shea, and Keeley Crockett. Sentence similarity based on semantic nets and corpus statistics. *IEEE Transactions on Knowledge and Data Engineering*, 18(8):1138–1150, 2006.
- Kenneth C. Litkowski. Sense information for disambiguation: Confluence of supervised and unsupervised methods. In *Proceedings of the ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pages 47–53. Association for Computational Linguistics, July 2002.
- Rada Mihalcea, Paul Tarau, and Elizabeth Figa. PageRank on semantic networks, with application to word sense disambiguation. In *Proceedings of the 20th International Conference on Computational Linguistics*, 2004.
- Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. Using distributional similarity for lexical expansion in knowledge-based word sense disambiguation. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 1781–1796, 2012.
- Roberto Navigli. Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):10:1–10:69, 2009.
- Siddharth Patwardhan and Ted Pedersen. Using WordNet-based context vectors to estimate the semantic relatedness of concepts. *Proceedings of the EACL 2006 Workshop Making Sense of Sense-Bringing Computational Linguistics and Psycholinguistics Together*, 1501:1–8, 2006.
- Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the 4th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 241–257, 2003.
- Ted Pedersen. A simple approach to building ensembles of Naive Bayesian classifiers for word sense disambiguation. In *Proceedings of the 1st Conference of North American Chapter of the Association for Computational Linguistics*, pages 63–69. Association for Computational Linguistics, 2000.
- Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. SemEval-2007 task 17: English lexical sample, SRL and all words. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 87–92. Association for Computational Linguistics, 2007.
- Ariel Raviv, Shaul Markovitch, and Sotirios-Efstathios Maneas. Concept-based approach to word sense disambiguation. In *Proceedings of the 26th Conference on Artificial Intelligence*, 2012.
- Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1, IJCAI’95*, pages 448–453, San Francisco, CA, USA, 1995.
- David Tugwell and Adam Kilgarriff. Wasp-bench: a lexicographic tool supporting word sense disambiguation. In *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 151–154. Association for Computational Linguistics, 2001.
- Peter Turney. Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the 12th European Conference on Machine Learning*, pages 491–502, 2001.
- Florentina Vasilescu, Philippe Langlais, and Guy Lapalme. Evaluating variants of the Lesk approach for disambiguating words. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, 2004.

Yorick Wilks, Dan Fass, Cheng-Ming Guo, James E. McDonald, Tony Plate, and Brian M. Slator. Providing machine tractable dictionary tools. *Machine Translation*, 5(2):99–154, 1990.

David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196, 1995.

Fast Easy Unsupervised Domain Adaptation with Marginalized Structured Dropout

Yi Yang and Jacob Eisenstein
School of Interactive Computing
Georgia Institute of Technology
{yiyang, jacob}@gatech.edu

Abstract

Unsupervised domain adaptation often relies on transforming the instance representation. However, most such approaches are designed for bag-of-words models, and ignore the structured features present in many problems in NLP. We propose a new technique called **marginalized structured dropout**, which exploits feature structure to obtain a remarkably simple and efficient feature projection. Applied to the task of fine-grained part-of-speech tagging on a dataset of historical Portuguese, marginalized structured dropout yields state-of-the-art accuracy while increasing speed by more than an order-of-magnitude over previous work.

1 Introduction

Unsupervised domain adaptation is a fundamental problem for natural language processing, as we hope to apply our systems to datasets unlike those for which we have annotations. This is particularly relevant as labeled datasets become stale in comparison with rapidly evolving social media writing styles (Eisenstein, 2013), and as there is increasing interest in natural language processing for historical texts (Piotrowski, 2012). While a number of different approaches for domain adaptation have been proposed (Pan and Yang, 2010; Søgaard, 2013), they tend to emphasize bag-of-words features for classification tasks such as sentiment analysis. Consequently, many approaches rely on each instance having a relatively large number of active features, and fail to exploit the structured feature spaces that characterize syntactic tasks such as sequence labeling and parsing (Smith, 2011).

As we will show, substantial efficiency improvements can be obtained by designing domain

adaptation methods for learning in structured feature spaces. We build on work from the deep learning community, in which *denoising autoencoders* are trained to remove synthetic noise from the observed instances (Glorot et al., 2011a). By using the autoencoder to transform the original feature space, one may obtain a representation that is less dependent on any individual feature, and therefore more robust across domains. Chen et al. (2012) showed that such autoencoders can be learned even as the noising process is analytically marginalized; the idea is similar in spirit to feature noising (Wang et al., 2013). While the marginalized denoising autoencoder (mDA) is considerably faster than the original denoising autoencoder, it requires solving a system of equations that can grow very large, as realistic NLP tasks can involve 10^5 or more features.

In this paper we investigate noising functions that are explicitly designed for *structured feature spaces*, which are common in NLP. For example, in part-of-speech tagging, Toutanova et al. (2003) define several feature “templates”: the current word, the previous word, the suffix of the current word, and so on. For each feature template, there are thousands of binary features. To exploit this structure, we propose two alternative noising techniques: (1) **feature scrambling**, which randomly chooses a feature template and randomly selects an alternative value within the template, and (2) **structured dropout**, which randomly eliminates all but a single feature template. We show how it is possible to marginalize over both types of noise, and find that the solution for structured dropout is substantially simpler and more efficient than the mDA approach of Chen et al. (2012), which does not consider feature structure.

We apply these ideas to fine-grained part-of-speech tagging on a dataset of Portuguese texts from the years 1502 to 1836 (Galves and Faria, 2010), training on recent texts and evaluating

on older documents. Both structure-aware domain adaptation algorithms perform as well as standard dropout — and better than the well-known structural correspondence learning (SCL) algorithm (Blitzer et al., 2007) — but structured dropout is more than an order-of-magnitude faster. As a secondary contribution of this paper, we demonstrate the applicability of unsupervised domain adaptation to the syntactic analysis of historical texts.

2 Model

In this section we first briefly describe the denoising autoencoder (Glorot et al., 2011b), its application to domain adaptation, and the analytic marginalization of noise (Chen et al., 2012). Then we present three versions of marginalized denoising autoencoders (mDA) by incorporating different types of noise, including two new noising processes that are designed for structured features.

2.1 Denoising Autoencoders

Assume instances $\mathbf{x}_1, \dots, \mathbf{x}_n$, which are drawn from both the source and target domains. We will “corrupt” these instances by adding different types of noise, and denote the corrupted version of \mathbf{x}_i by $\tilde{\mathbf{x}}_i$. Single-layer denoising autoencoders reconstruct the corrupted inputs with a projection matrix $\mathbf{W} : \mathbb{R}^d \rightarrow \mathbb{R}^d$, which is estimated by minimizing the squared reconstruction loss

$$\mathcal{L} = \frac{1}{2} \sum_{i=1}^n \|\mathbf{x}_i - \mathbf{W}\tilde{\mathbf{x}}_i\|^2. \quad (1)$$

If we write $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$, and we write its corrupted version $\tilde{\mathbf{X}}$, then the loss in (1) can be written as

$$\mathcal{L}(\mathbf{W}) = \frac{1}{2n} \text{tr} \left[\left(\mathbf{X} - \mathbf{W}\tilde{\mathbf{X}} \right)^\top \left(\mathbf{X} - \mathbf{W}\tilde{\mathbf{X}} \right) \right]. \quad (2)$$

In this case, we have the well-known closed-form solution for this ordinary least square problem:

$$\mathbf{W} = \mathbf{P}\mathbf{Q}^{-1}, \quad (3)$$

where $\mathbf{Q} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top$ and $\mathbf{P} = \mathbf{X}\tilde{\mathbf{X}}^\top$. After obtaining the weight matrix \mathbf{W} , we can insert non-linearity into the output of the denoiser, such as $\tanh(\mathbf{W}\tilde{\mathbf{X}})$. It is also possible to apply stacking, by passing this vector through another autoencoder (Chen et al., 2012). In pilot experiments, this slowed down estimation and had little effect on accuracy, so we did not include it.

High-dimensional setting Structured prediction tasks often have much more features than simple bag-of-words representation, and performance relies on the rare features. In a naive implementation of the denoising approach, both \mathbf{P} and \mathbf{Q} will be dense matrices with dimensionality $d \times d$, which would be roughly 10^{11} elements in our experiments. To solve this problem, Chen et al. (2012) propose to use a set of pivot features, and train the autoencoder to reconstruct the pivots from the full set of features. Specifically, the corrupted input is divided to S subsets $\tilde{\mathbf{x}}_i = \left[(\tilde{\mathbf{x}}_i^1)^\top, \dots, (\tilde{\mathbf{x}}_i^S)^\top \right]^\top$. We obtain a projection matrix \mathbf{W}^s for each subset by reconstructing the pivot features from the features in this subset; we can then use the sum of all reconstructions as the new features, $\tanh(\sum_{s=1}^S \mathbf{W}^s \mathbf{X}^s)$.

Marginalized Denoising Autoencoders In the standard denoising autoencoder, we need to generate multiple versions of the corrupted data $\tilde{\mathbf{X}}$ to reduce the variance of the solution (Glorot et al., 2011b). But Chen et al. (2012) show that it is possible to marginalize over the noise, analytically computing expectations of both \mathbf{P} and \mathbf{Q} , and computing

$$\mathbf{W} = E[\mathbf{P}]E[\mathbf{Q}]^{-1}, \quad (4)$$

where $E[\mathbf{P}] = \sum_{i=1}^n E[\mathbf{x}_i \tilde{\mathbf{x}}_i^\top]$ and $E[\mathbf{Q}] = \sum_{i=1}^n E[\tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^\top]$. This is equivalent to corrupting the data $m \rightarrow \infty$ times. The computation of these expectations depends on the type of noise.

2.2 Noise distributions

Chen et al. (2012) used dropout noise for domain adaptation, which we briefly review. We then describe two novel types of noise that are designed for structured feature spaces, and explain how they can be marginalized to efficiently compute \mathbf{W} .

Dropout noise In dropout noise, each feature is set to zero with probability $p > 0$. If we define the scatter matrix of the uncorrupted input as $\mathbf{S} = \mathbf{X}\mathbf{X}^\top$, the solutions under dropout noise are

$$E[\mathbf{Q}]_{\alpha,\beta} = \begin{cases} (1-p)^2 \mathbf{S}_{\alpha,\beta} & \text{if } \alpha \neq \beta \\ (1-p) \mathbf{S}_{\alpha,\beta} & \text{if } \alpha = \beta \end{cases}, \quad (5)$$

and

$$E[\mathbf{P}]_{\alpha,\beta} = (1-p) \mathbf{S}_{\alpha,\beta}, \quad (6)$$

where α and β index two features. The form of these solutions means that computing \mathbf{W} requires solving a system of equations equal to the number of features (in the naive implementation), or several smaller systems of equations (in the high-dimensional version). Note also that p is a tunable parameter for this type of noise.

Structured dropout noise In many NLP settings, we have several feature templates, such as previous-word, middle-word, next-word, etc, with only one feature per template firing on any token. We can exploit this structure by using an alternative dropout scheme: for each token, choose exactly one feature template to keep, and zero out all other features that consider this token (transition feature templates such as $\langle y_t, y_{t-1} \rangle$ are not considered for dropout). Assuming we have K feature templates, this noise leads to very simple solutions for the marginalized matrices $E[\mathbf{P}]$ and $E[\mathbf{Q}]$,

$$E[\mathbf{Q}]_{\alpha,\beta} = \begin{cases} 0 & \text{if } \alpha \neq \beta \\ \frac{1}{K} \mathbf{S}_{\alpha,\beta} & \text{if } \alpha = \beta \end{cases} \quad (7)$$

$$E[\mathbf{P}]_{\alpha,\beta} = \frac{1}{K} \mathbf{S}_{\alpha,\beta}, \quad (8)$$

For $E[\mathbf{P}]$, we obtain a scaled version of the scatter matrix, because in each instance $\tilde{\mathbf{x}}$, there is exactly a $1/K$ chance that each individual feature survives dropout. $E[\mathbf{Q}]$ is diagonal, because for any off-diagonal entry $E[\mathbf{Q}]_{\alpha,\beta}$, at least one of α and β will drop out for every instance. We can therefore view the projection matrix \mathbf{W} as a row-normalized version of the scatter matrix \mathbf{S} . Put another way, the contribution of β to the reconstruction for α is equal to the co-occurrence count of α and β , divided by the count of β .

Unlike standard dropout, there are no free hyper-parameters to tune for structured dropout. Since $E[\mathbf{Q}]$ is a diagonal matrix, we eliminate the cost of matrix inversion (or of solving a system of linear equations). Moreover, to extend mDA for high dimensional data, we no longer need to divide the corrupted input $\tilde{\mathbf{x}}$ to several subsets.¹

For intuition, consider standard feature dropout with $p = \frac{K-1}{K}$. This will look very similar to structured dropout: the matrix $E[\mathbf{P}]$ is identical, and $E[\mathbf{Q}]$ has off-diagonal elements which are scaled by $(1-p)^2$, which goes to zero as K is

¹ $E[\mathbf{P}]$ is an r by d matrix, where r is the number of pivots.

large. However, by including these elements, standard dropout is considerably slower, as we show in our experiments.

Scrambling noise A third alternative is to “scramble” the features by randomly selecting alternative features within each template. For a feature α belonging to a template F , with probability p we will draw a noise feature β also belonging to F , according to some distribution q . In this work, we use an uniform distribution, in which $q_\beta = \frac{1}{|F|}$. However, the below solutions will also hold for other scrambling distributions, such as mean-preserving distributions.

Again, it is possible to analytically marginalize over this noise. Recall that $E[\mathbf{Q}] = \sum_{i=1}^n E[\tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^\top]$. An off-diagonal entry in the matrix $\tilde{\mathbf{x}} \tilde{\mathbf{x}}^\top$ which involves features α and β belonging to different templates ($F_\alpha \neq F_\beta$) can take four different values ($\mathbf{x}_{i,\alpha}$ denotes feature α in \mathbf{x}_i):

- $\mathbf{x}_{i,\alpha} \mathbf{x}_{i,\beta}$ if both features are unchanged, which happens with probability $(1-p)^2$.
- 1 if both features are chosen as noise features, which happens with probability $p^2 q_\alpha q_\beta$.
- $\mathbf{x}_{i,\alpha}$ or $\mathbf{x}_{i,\beta}$ if one feature is unchanged and the other one is chosen as the noise feature, which happens with probability $p(1-p)q_\beta$ or $p(1-p)q_\alpha$.

The diagonal entries take the first two values above, with probability $1-p$ and $p q_\alpha$ respectively. Other entries will be all zero (only one feature belonging to the same template will fire in \mathbf{x}_i). We can use similar reasoning to compute the expectation of \mathbf{P} . With probability $(1-p)$, the original features are preserved, and we add the outer-product $\mathbf{x}_i \mathbf{x}_i^\top$; with probability p , we add the outer-product $\mathbf{x}_i q^\top$. Therefore $E[\mathbf{P}]$ can be computed as the sum of these terms.

3 Experiments

We compare these methods on historical Portuguese part-of-speech tagging, creating domains over historical epochs.

3.1 Experiment setup

Datasets We use the Tycho Brahe corpus to evaluate our methods. The corpus contains a total of 1,480,528 manually tagged words. It uses a set of 383 tags and is composed of various texts from

historical Portuguese, from 1502 to 1836. We divide the texts into fifty-year periods to create different domains. Table 1 presents some statistics of the datasets. We hold out 5% of data as development data to tune parameters. The two most recent domains (1800-1849 and 1750-1849) are treated as source domains, and the other domains are target domains. This scenario is motivated by training a tagger on a modern newstext corpus and applying it to historical documents.

Dataset	# of Tokens				
	Total	Narrative	Letters	Dissertation	Theatre
1800-1849	125719	91582	34137	0	0
1750-1799	202346	57477	84465	0	60404
1700-1749	278846	0	130327	148519	0
1650-1699	248194	83938	115062	49194	0
1600-1649	295154	117515	115252	62387	0
1550-1599	148061	148061	0	0	0
1500-1549	182208	126516	0	55692	0
Overall	1480528	625089	479243	315792	60404

Table 1: Statistics of the Tycho Brahe Corpus

CRF tagger We use a conditional random field tagger, choosing CRFsuite because it supports arbitrary real valued features (Okazaki, 2007), with SGD optimization. Following the work of Nogueira Dos Santos et al. (2008) on this dataset, we apply the feature set of Ratnaparkhi (1996). There are 16 feature templates and 372,902 features in total. Following Blitzer et al. (2006), we consider pivot features that appear more than 50 times in all the domains. This leads to a total of 1572 pivot features in our experiments.

Methods We compare mDA with three alternative approaches. We refer to *baseline* as training a CRF tagger on the source domain and testing on the target domain with only base features. We also include *PCA* to project the entire dataset onto a low-dimensional sub-space (while still including the original features). Finally, we compare against Structural Correspondence Learning (*SCL*; Blitzer et al., 2006), another feature learning algorithm. In all cases, we include the entire dataset to compute the feature projections; we also conducted experiments using only the test and training data for feature projections, with very similar results.

Parameters All the hyper-parameters are decided with our development data on the training set. We try different low dimension K from 10 to

2000 for PCA. Following Blitzer (2008) we perform feature centering/normalization, as well as rescaling for SCL. The best parameters for SCL are dimensionality $K = 25$ and rescale factor $\alpha = 5$, which are the same as in the original paper. For mDA, the best corruption level is $p = 0.9$ for dropout noise, and $p = 0.1$ for scrambling noise. Structured dropout noise has no free hyper-parameters.

3.2 Results

Table 2 presents results for different domain adaptation tasks. We also compute the *transfer ratio*, which is defined as $\frac{\text{adaptation accuracy}}{\text{baseline accuracy}}$, shown in Figure 1. The generally positive trend of these graphs indicates that adaptation becomes progressively more important as we select test sets that are more temporally remote from the training data.

In general, mDA outperforms SCL and PCA, the latter of which shows little improvement over the base features. The various noising approaches for mDA give very similar results. However, structured dropout is orders of magnitude faster than the alternatives, as shown in Table 3. The scrambling noise is most time-consuming, with cost dominated by a matrix multiplication.

Method	PCA	SCL	mDA		
			dropout	structured	scrambling
Time	7,779	38,849	8,939	339	327,075

Table 3: Time, in seconds, to compute the feature transformation

4 Related Work

Domain adaptation Most previous work on domain adaptation focused on the supervised setting, in which some labeled data is available in the target domain (Jiang and Zhai, 2007; Daumé III, 2007; Finkel and Manning, 2009). Our work focuses on unsupervised domain adaptation, where no labeled data is available in the target domain. Several representation learning methods have been proposed to solve this problem. In structural correspondence learning (SCL), the induced representation is based on the task of predicting the presence of pivot features. Autoencoders apply a similar idea, but use the denoised instances as the latent representation (Vincent et al., 2008; Glorot et al., 2011b; Chen et al., 2012). Within the context of denoising autoencoders, we have focused

Task	baseline	PCA	SCL	mDA		
				dropout	structured	scrambling
from 1800-1849						
→ 1750	89.12	89.09	89.69	90.08	90.08	90.01
→ 1700	90.43	90.43	91.06	91.56	91.57	91.55
→ 1650	88.45	88.52	87.09	88.69	88.70	88.57
→ 1600	87.56	87.58	88.47	89.60	89.61	89.54
→ 1550	89.66	89.61	90.57	91.39	91.39	91.36
→ 1500	85.58	85.63	86.99	88.96	88.95	88.91
from 1750-1849						
→ 1700	94.64	94.62	94.81	95.08	95.08	95.02
→ 1650	91.98	90.97	90.37	90.83	90.84	90.80
→ 1600	92.95	92.91	93.17	93.78	93.78	93.71
→ 1550	93.27	93.21	93.75	94.06	94.05	94.02
→ 1500	89.80	89.75	90.59	91.71	91.71	91.68

Table 2: Accuracy results for adaptation from labeled data in 1800-1849, and in 1750-1849.

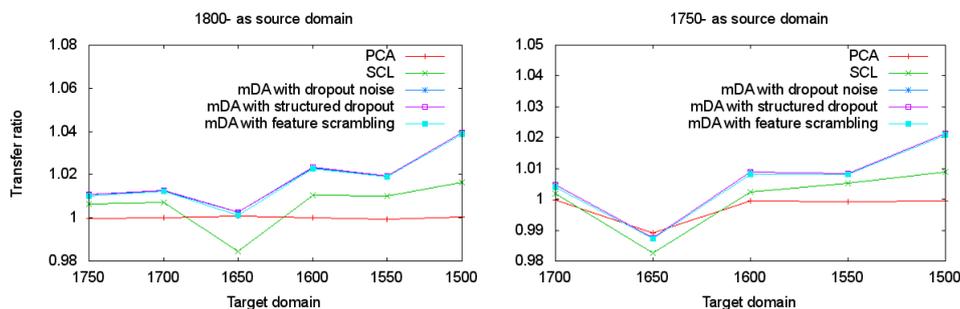


Figure 1: Transfer ratio for adaptation to historical text

on dropout noise, which has also been applied as a general technique for improving the robustness of machine learning, particularly in neural networks (Hinton et al., 2012; Wang et al., 2013).

On the specific problem of sequence labeling, Xiao and Guo (2013) proposed a supervised domain adaptation method by using a log-bilinear language adaptation model. Dhillon et al. (2011) presented a spectral method to estimate low dimensional context-specific word representations for sequence labeling. Huang and Yates (2009; 2012) used an HMM model to learn latent representations, and then leverage the Posterior Regularization framework to incorporate specific biases. Unlike these methods, our approach uses a standard CRF, but with transformed features.

Historical text Our evaluation concerns syntactic analysis of historical text, which is a topic of increasing interest for NLP (Piotrowski, 2012). Penacchiotti and Zanzotto (2008) find that part-of-speech tagging degrades considerably when applied to a corpus of historical Italian. Moon and Baldrige (2007) tackle the challenging problem of tagging Middle English, using techniques for

projecting syntactic annotations across languages. Prior work on the Tycho Brahe corpus applied supervised learning to a random split of test and training data (Kepler and Finger, 2006; Dos Santos et al., 2008); they did not consider the domain adaptation problem of training on recent data and testing on older historical text.

5 Conclusion and Future Work

Denosing autoencoders provide an intuitive solution for domain adaptation: transform the features into a representation that is resistant to the noise that may characterize the domain adaptation process. The original implementation of this idea produced this noise directly (Glorot et al., 2011b); later work showed that dropout noise could be analytically marginalized (Chen et al., 2012). We take another step towards simplicity by showing that structured dropout can make marginalization even easier, obtaining dramatic speedups without sacrificing accuracy.

Acknowledgments : We thank the reviewers for useful feedback. This research was supported by National Science Foundation award 1349837.

References

- 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 '06*, pages 120–128, Stroudsburg, PA, USA. Association for Computational Linguistics.
- John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Association for Computational Linguistics*, Prague, Czech Republic.
- John Blitzer. 2008. *Domain Adaptation of Natural Language Processing Systems*. Ph.D. thesis, University of Pennsylvania.
- Minmin Chen, Zhixiang Xu, Kilian Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ICML '12, pages 767–774. ACM, New York, NY, USA, July.
- Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *ACL*, volume 1785, page 1787.
- Paramveer S Dhillon, Dean P Foster, and Lyle H Ungar. 2011. Multi-view learning of word embeddings via cca. In *NIPS*, volume 24, pages 199–207.
- Cícero Nogueira Dos Santos, Ruy L Milidiú, and Raúl P Rentería. 2008. Portuguese part-of-speech tagging using entropy guided transformation learning. In *Computational Processing of the Portuguese Language*, pages 143–152. Springer.
- Jacob Eisenstein. 2013. What to do about bad language on the internet. In *Proceedings of NAACL*, Atlanta, GA.
- Jenny Rose Finkel and Christopher D Manning. 2009. Hierarchical bayesian domain adaptation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 602–610. Association for Computational Linguistics.
- Charlotte Galves and Pablo Faria. 2010. Tycho Brahe Parsed Corpus of Historical Portuguese. <http://www.tycho.iel.unicamp.br/tycho/corpus/en/index.html>.
- Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011a. Deep sparse rectifier networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP Volume*, volume 15, pages 315–323.
- Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011b. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 513–520.
- Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.
- Fei Huang and Alexander Yates. 2009. Distributional representations for handling sparsity in supervised sequence-labeling. 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: Volume 1-Volume 1*, pages 495–503. Association for Computational Linguistics.
- Fei Huang and Alexander Yates. 2012. Biased representation learning for domain adaptation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1313–1323. Association for Computational Linguistics.
- Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in nlp. In *ACL*, volume 2007, page 22.
- Fábio N Kepler and Marcelo Finger. 2006. Comparing two markov methods for part-of-speech tagging of portuguese. In *Advances in Artificial Intelligence-IBERAMIA-SBIA 2006*, pages 482–491. Springer.
- Taesun Moon and Jason Baldrige. 2007. Part-of-speech tagging for middle english through alignment and projection of parallel diachronic texts. In *EMNLP-CoNLL*, pages 390–399.
- Cícero Nogueira Dos Santos, Ruy L. Milidiú, and Raúl P. Rentería. 2008. Portuguese part-of-speech tagging using entropy guided transformation learning. In *Proceedings of the 8th international conference on Computational Processing of the Portuguese Language, PROPOR '08*, pages 143–152, Berlin, Heidelberg. Springer-Verlag.
- Naoaki Okazaki. 2007. Crfsuite: a fast implementation of conditional random fields (crfs).
- Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359.
- Marco Pennacchiotti and Fabio Massimo Zanzotto. 2008. Natural language processing across time: An empirical investigation on italian. In *Advances in Natural Language Processing*, pages 371–382. Springer.
- Michael Piotrowski. 2012. Natural language processing for historical texts. *Synthesis Lectures on Human Language Technologies*, 5(2):1–157.

- Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, April 16.
- Noah A Smith. 2011. Linguistic structure prediction. *Synthesis Lectures on Human Language Technologies*, 4(2):1–274.
- Anders Søgaard. 2013. Semi-supervised learning and domain adaptation in natural language processing. *Synthesis Lectures on Human Language Technologies*, 6(2):1–103.
- Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology—Volume 1*, pages 173–180. Association for Computational Linguistics.
- Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM.
- Sida I. Wang, Mengqiu Wang, Stefan Wager, Percy Liang, and Christopher D. Manning. 2013. Feature noising for log-linear structured prediction. In *Empirical Methods in Natural Language Processing (EMNLP)*.
- Min Xiao and Yuhong Guo. 2013. Domain adaptation for sequence labeling tasks with a probabilistic language adaptation model. In Sanjoy Dasgupta and David Mcallester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, volume 28, pages 293–301. JMLR Workshop and Conference Proceedings.

Improving Lexical Embeddings with Semantic Knowledge

Mo Yu *

Machine Translation Lab
Harbin Institute of Technology
Harbin, China
gflfof@gmail.com

Mark Dredze

Human Language Technology Center of Excellence
Center for Language and Speech Processing
Johns Hopkins University
Baltimore, MD 21218
mdredze@cs.jhu.edu

Abstract

Word embeddings learned on unlabeled data are a popular tool in semantics, but may not capture the desired semantics. We propose a new learning objective that incorporates both a neural language model objective (Mikolov et al., 2013) and prior knowledge from semantic resources to learn improved lexical semantic embeddings. We demonstrate that our embeddings improve over those learned solely on raw text in three settings: language modeling, measuring semantic similarity, and predicting human judgements.

1 Introduction

Word embeddings are popular representations for syntax (Turian et al., 2010; Collobert and Weston, 2008; Mnih and Hinton, 2007), semantics (Huang et al., 2012; Socher et al., 2013), morphology (Luong et al., 2013) and other areas. A long line of embeddings work, such as LSA and randomized embeddings (Ravichandran et al., 2005; Van Durme and Lall, 2010), has recently turned to neural language models (Bengio et al., 2006; Collobert and Weston, 2008; Turian et al., 2010). Unsupervised learning can take advantage of large corpora, which can produce impressive results.

However, the main drawback of unsupervised learning is that the learned embeddings may not be suited for the task of interest. Consider semantic embeddings, which may capture a notion of semantics that improves one semantic task but harms another. Controlling this behavior is challenging with an unsupervised objective. However, rich prior knowledge exists for many tasks, and there are numerous such semantic resources.

We propose a new training objective for learning word embeddings that incorporates prior

knowledge. Our model builds on word2vec (Mikolov et al., 2013), a neural network based language model that learns word embeddings by maximizing the probability of raw text. We extend the objective to include prior knowledge about synonyms from semantic resources; we consider both the Paraphrase Database (Ganitkevitch et al., 2013) and WordNet (Fellbaum, 1999), which annotate semantic relatedness between words. The latter was also used in (Bordes et al., 2012) for training a network for predicting synset relation. The combined objective maximizes both the probability of the raw corpus and encourages embeddings to capture semantic relations from the resources. We demonstrate improvements in our embeddings on three tasks: language modeling, measuring word similarity, and predicting human judgements on word pairs.

2 Learning Embeddings

We present a general model for learning word embeddings that incorporates prior knowledge available for a domain. While in this work we consider semantics, our model could incorporate prior knowledge from many types of resources. We begin by reviewing the word2vec objective and then present augmentations of the objective for prior knowledge, including different training strategies.

2.1 Word2vec

Word2vec (Mikolov et al., 2013) is an algorithm for learning embeddings using a neural language model. Embeddings are represented by a set of latent (hidden) variables, and each word is represented by a specific instantiation of these variables. Training learns these representations for each word w_t (the t th word in a corpus of size T) so as to maximize the log likelihood of each token given its context: words within a window sized c :

$$\max \frac{1}{T} \sum_{t=1}^T \log p(w_t | w_{t-c}^{t+c}), \quad (1)$$

This work was done while the author was visiting JHU.

where w_{t-c}^{t+c} is the set of words in the window of size c centered at w_t (w_t excluded).

Word2vec offers two choices for modeling of Eq. (1): a skip-gram model and a continuous bag-of-words model (cbow). The latter worked better in our experiments so we focus on it in our presentation. cbow defines $p(w_t|w_{t-c}^{t+c})$ as:

$$\frac{\exp\left(e'_{w_t} \cdot \sum_{-c \leq j \leq c, j \neq 0} e_{w_{t+j}}\right)}{\sum_w \exp\left(e'_w \cdot \sum_{-c \leq j \leq c, j \neq 0} e_{w_{t+j}}\right)}, \quad (2)$$

where e_w and e'_w represent the input and output embeddings respectively, i.e., the assignments to the latent variables for word w . While some learn a single representation for each word ($e'_w \triangleq e_w$), our results improved when we used a separate embedding for input and output in cbow.

2.2 Relation Constrained Model

Suppose we have a resource that indicates relations between words. In the case of semantics, we could have a resource that encodes semantic similarity between words. Based on this resource, we learn embeddings that predict one word from another related word. We define \mathbf{R} as a set of relations between two words w and w' . \mathbf{R} can contain typed relations (e.g., w is related to w' through a specific type of semantic relation), and relations can have associated scores indicating their strength. We assume a single relation type of uniform strength, though it is straightforward to include additional characteristics into the objective.

Define \mathbf{R}_w to be the subset of relations in \mathbf{R} which involve word w . Our objective maximizes the (log) probability of all relations by summing over all words N in the vocabulary:

$$\frac{1}{N} \sum_{i=1}^N \sum_{w \in \mathbf{R}_{w_i}} \log p(w|w_i), \quad (3)$$

$p(w|w_i) = \exp\left(e'_w \cdot e_{w_i}\right) / \sum_{\bar{w}} \exp\left(e'_{\bar{w}} \cdot e_{w_i}\right)$ takes a form similar to Eq. (2) but without the context: e and e' are again the input and output embeddings. For our semantic relations e'_w and e_w are symmetrical, so we use a single embedding. Embeddings are learned such that they are predictive of related words in the resource. We call this the Relation Constrained Model (RCM).

2.3 Joint Model

The cbow and RCM objectives use separate data for learning. While RCM learns embeddings

suited to specific tasks based on knowledge resources, cbow learns embeddings for words not included in the resource but appear in a corpus. We form a joint model through a linear combination of the two (weighted by C):

$$\frac{1}{T} \sum_{t=1}^T \log p(w_t|w_{t-c}^{t+c}) + \frac{C}{N} \sum_{i=1}^N \sum_{w \in \mathbf{R}_{w_i}} \log p(w|w_i)$$

Based on our initial experiments, RCM uses the output embeddings of cbow.

We learn embeddings using stochastic gradient ascent. Updates for the first term for e' and e are:

$$e'_w - \alpha_{\text{cbow}} \left(\sigma(f(w)) - I_{[w=w_t]} \right) \cdot \sum_{j=t-c}^{t+c} e_{w_j}$$

$$e_{w_j} - \alpha_{\text{cbow}} \sum_w \left(\sigma(f(w)) - I_{[w=w_t]} \right) \cdot e'_w,$$

where $\sigma(x) = \exp\{x\} / (1 + \exp\{x\})$, $I_{[x]}$ is 1 when x is true, $f(w) = e'_w \cdot \sum_{j=t-c}^{t+c} e_{w_j}$. Second term updates are:

$$e'_w - \alpha_{\text{RCM}} \left(\sigma(f'(w)) - I_{[w \in \mathbf{R}_{w_i}]} \right) \cdot e'_{w_i}$$

$$e'_{w_i} - \alpha_{\text{RCM}} \sum_w \left(\sigma(f'(w)) - I_{[w \in \mathbf{R}_{w_i}]} \right) \cdot e'_w,$$

where $f'(w) = e'_w \cdot e'_{w_i}$. We use two learning rates: α_{cbow} and α_{RCM} .

2.4 Parameter Estimation

All three models (cbow, RCM and joint) use the same training scheme based on Mikolov et al. (2013). There are several choices to make in parameter estimation; we present the best performing choices used in our results.

We use noise contrastive estimation (NCE) (Mnih and Teh, 2012), which approximately maximizes the log probability of the softmax objective (Eq. 2). For each objective (cbow or RCM), we sample 15 words as negative samples for each training instance according to their frequencies in raw texts (i.e. training data of cbow). Suppose w has frequency $u(w)$, then the probability of sampling w is $p(w) \propto u(w)^{3/4}$.

We use distributed training, where shared embeddings are updated by each thread based on training data within the thread, i.e., asynchronous stochastic gradient ascent. For the joint model, we assign threads to the cbow or RCM objective with a balance of 12:1 (i.e. C is approximately $\frac{1}{12}$). We allow the cbow threads to control convergence; training stops when these threads finish processing the data. We found this an effective method

for balancing the two objectives. We trained each cbow objective using a single pass over the data set (except for those in Section 4.1), which we empirically verified was sufficient to ensure stable performances on semantic tasks.

Model pre-training is critical in deep learning (Bengio et al., 2007; Erhan et al., 2010). We evaluate two strategies: random initialization, and pre-training the embeddings. For pre-training, we first learn using cbow with a random initialization. The resulting trained model is then used to initialize the RCM model. This enables the RCM model to benefit from the unlabeled data, but refine the embeddings constrained by the given relations.

Finally, we consider a final model for training embeddings that uses a specific training regime. While the joint model balances between fitting the text and learning relations, modeling the text at the expense of the relations may negatively impact the final embeddings for tasks that use the embeddings outside of the context of word2vec. Therefore, we use the embeddings from a trained joint model to pre-train an RCM model. We call this setting Joint→RCM.

3 Evaluation

For training cbow we use the New York Times (NYT) 1994-97 subset from Gigaword v5.0 (Parker et al., 2011). We select 1,000 paragraphs each for dev and test data from the December 2010 portion of the NYT. Sentences are tokenized using OpenNLP¹, yielding 518,103,942 tokens for training, 42,953 tokens for dev and 41,344 for test.

We consider two resources for training the RCM term: the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013) and WordNet (Fellbaum, 1999). For each semantic pair extracted from these resources, we add a relation to the RCM objective. Since we use both resources for evaluation, we divide each into train, dev and test.

PPDB is an automatically extracted dataset containing tens of millions of paraphrase pairs, including words and phrases. We used the “lexical” version of PPDB (no phrases) and filtered to include pairs that contained words found in the 200,000 most frequent words in the NYT corpus, which ensures each word in the relations had support in the text corpus. Next, we removed duplicate pairs: if $\langle A, B \rangle$ occurred in PPDB, we removed relations of $\langle B, A \rangle$. PPDB is organized

¹<https://opennlp.apache.org/>

PPDB		Relations	WordNet	Relations
Train	XL	115,041	Train (not used in this work)	68,372
	XXL	587,439		
	XXXL	2,647,105		
Dev		1,582	Dev	1,500
Test		1,583	Test	1,500

Table 1: Sizes of semantic resources datasets.

into 6 parts, ranging from S (small) to XXXL. Division into these sets is based on an automatically derived accuracy metric. Since S contains the most accurate paraphrases, we used these for evaluation. We divided S into a dev set (1582 pairs) and test set (1583 pairs). Training was based on one of the other sets minus relations from S.

We created similar splits using WordNet, extracting synonyms using the 100,000 most frequent NYT words. We divide the vocabulary into three sets: the most frequent 10,000 words, words with ranks between 10,001-30,000 and 30,001-100,000. We sample 500 words from each set to construct a dev and test set. For each word we sample one synonym to form a pair. The remaining words and their synonyms are used for training. However we did not use the training data because it is too small to affect the results. Table 1 summarizes the datasets.

4 Experiments

The goal of our experiments is to demonstrate the value of learning semantic embeddings with information from semantic resources. In each setting, we will compare the word2vec baseline embedding trained with cbow against RCM alone, the joint model and Joint→RCM. We consider three evaluation tasks: language modeling, measuring semantic similarity, and predicting human judgments on semantic relatedness. In all of our experiments, we conducted model development and tuned model parameters (C , α_{cbow} , α_{RCM} , PPDB dataset, etc.) on development data, and evaluate the best performing model on test data. The models are notated as follows: word2vec for the baseline objective (cbow or skip-gram), RCM-r/p and Joint-r/p for random and pre-trained initializations of the RCM and Joint objectives, and Joint→RCM for pre-training RCM with Joint embeddings. Unless otherwise notes, we train using PPDB XXL. We initially created WordNet training data, but found it too small to affect results. Therefore, we include only RCM results *trained* on PPDB, but show evaluations on both PPDB and WordNet.

Model	NCE	HS
word2vec (cbow)	8.75	6.90
RCM-p	8.55	7.07
Joint-r ($\alpha_{RCM} = 1 \times 10^{-2}$)	8.33	6.87
Joint-r ($\alpha_{RCM} = 1 \times 10^{-3}$)	8.20	6.75
Joint→RCM	8.40	6.92

Table 2: LM evaluation on held out NYT data.

We trained 200-dimensional embeddings and used output embeddings for measuring similarity. During the training of cbow objectives we remove all words with frequencies less than 5, which is the default setting of word2vec.

4.1 Language Modeling

Word2vec is fundamentally a language model, which allows us to compute standard evaluation metrics on a held out dataset. After obtaining trained embeddings from any of our objectives, we use the embeddings in the word2vec model to measure perplexity of the test set. Measuring perplexity means computing the exact probability of each word, which requires summation over all words in the vocabulary in the denominator of the softmax. Therefore, we also trained the language models with hierarchical classification (Mikolov et al., 2013) strategy (HS). The averaged perplexities are reported on the NYT test set.

While word2vec and joint are trained as language models, RCM is not. In fact, RCM does not even observe all the words that appear in the training set, so it makes little sense to use the RCM embeddings directly for language modeling. Therefore, in order to make fair comparison, for every set of trained embeddings, we fix them as input embedding for word2vec, then learn the remaining input embeddings (words not in the relations) and all the output embeddings using cbow. Since this involves running cbow on NYT data for 2 iterations (one iteration for word2vec-training/pre-training/joint-modeling and the other for tuning the language model), we use Joint-r (random initialization) for a fair comparison.

Table 2 shows the results for language modeling on test data. All of our proposed models improve over the baseline in terms of perplexity when NCE is used for training LMs. When HS is used, the perplexities are greatly improved. However in this situation only the joint models improve the results; and Joint→RCM performs similar to the baseline, although it is not designed for language modeling. We include the optimal α_{RCM}

in the table; while set $\alpha_{cbow} = 0.025$ (the default setting of word2vec). Even when our goal is to strictly model the raw text corpus, we obtain improvements by injecting semantic information into the objective. RCM can effectively shift learning to obtain more informative embeddings.

4.2 Measuring Semantic Similarity

Our next task is to find semantically related words using the embeddings, evaluating on relations from PPDB and WordNet. For each of the word pairs in the evaluation set $\langle A, B \rangle$, we use the cosine distance between the embeddings to score A with a candidate word B' . We use a large sample of candidate words (10k, 30k or 100k) and rank all candidate words for pairs where B appears in the candidates. We then measure the rank of the correct B to compute mean reciprocal rank (MRR). Our goal is to use word A to select word B as the closest matching word from the large set of candidates. Using this strategy, we evaluate the embeddings from all of our objectives and measure which embedding most accurately selected the true correct word.

Table 3 shows MRR results for both PPDB and WordNet dev and test datasets for all models. All of our methods improve over the baselines in nearly every test set result. In nearly every case, Joint→RCM obtained the largest improvements. Clearly, our embeddings are much more effective at capturing semantic similarity.

4.3 Human Judgements

Our final evaluation is to predict human judgements of semantic relatedness. We have pairs of words from PPDB scored by annotators on a scale of 1 to 5 for quality of similarity. Our data are the judgements used by Ganitkevitch et al. (2013), which we filtered to include only those pairs for which we learned embeddings, yielding 868 pairs.

We assign a score using the dot product between the output embeddings of each word in the pair, then order all 868 pairs according to this score. Using the human judgements, we compute the swapped pairs rate: the ratio between the number of swapped pairs and the number of all pairs. For pair p scored y_p by the embeddings and judged \hat{y}_p by an annotator, the swapped pair rate is:

$$\frac{\sum_{p_1, p_2 \in D} \mathbb{I}[(y_{p_1} - y_{p_2})(\hat{y}_{p_2} - \hat{y}_{p_1}) < 0]}{\sum_{p_1, p_2 \in D} \mathbb{I}[y_{p_1} \neq y_{p_2}]} \quad (4)$$

where $\mathbb{I}[x]$ is 1 when x is true.

Model	PPDB						WordNet					
	Dev			Test			Dev			Test		
	10k	30k	100k	10k	30k	100k	10k	30k	100k	10k	30k	100k
word2vec (cbow)	49.68	39.26	29.15	49.31	42.53	30.28	10.24	8.64	5.14	10.04	7.90	4.97
word2vec (skip-gram)	48.70	37.14	26.20	-	-	-	8.61	8.10	4.62	-	-	-
RCM-r	55.03	42.52	26.05	-	-	-	13.33	9.05	5.29	-	-	-
RCM-p	61.79	53.83	40.95	65.42	55.82	41.20	15.25	12.13	7.46	14.13	11.23	7.39
Joint-r	59.91	50.87	36.81	-	-	-	15.73	11.36	7.14	13.97	10.51	7.44
Joint-p	59.75	50.93	37.73	64.30	53.27	38.97	15.61	11.20	6.96	-	-	-
Joint→RCM	64.22	54.99	41.34	68.20	57.87	42.64	16.81	11.67	7.55	16.16	11.21	7.56

Table 3: MRR for semantic similarity on PPDB and WordNet dev and test data. Higher is better. All RCM objectives are trained with PPDB XXL. To preserve test data integrity, only the best performing setting of each model is evaluated on the test data.

Model	Swapped Pairs Rate
word2vec (cbow)	17.81
RCM-p	16.66
Joint-r	16.85
Joint-p	16.96
Joint→RCM	16.62

Table 4: Results for ranking the quality of PPDB pairs as compared to human judgements.

Model	Relations	PPDB Dev		
		10k	30k	100k
RCM-r	XL	24.02	15.26	9.55
RCM-p	XL	54.97	45.35	32.95
RCM-r	XXL	55.03	42.52	26.05
RCM-p	XXL	61.79	53.83	40.95
RCM-r	XXXL	51.00	44.61	28.42
RCM-p	XXXL	53.01	46.35	34.19

Table 5: MRR on PPDB dev data for training on an increasing number of relations.

Table 4 shows that all of our models obtain reductions in error as compared to the baseline (cbow), with Joint→RCM obtaining the largest reduction. This suggests that our embeddings are better suited for semantic tasks, in this case judged by human annotations.

Model	α_{RCM}	PPDB Dev		
		10k	30k	100k
Joint-p	1×10^{-1}	47.17	36.74	24.50
	5×10^{-2}	54.31	44.52	33.07
	1×10^{-2}	59.75	50.93	37.73
	1×10^{-3}	57.00	46.84	34.45

Table 6: Effect of learning rate α_{RCM} on MRR for the RCM objective in Joint models.

4.4 Analysis

We conclude our experiments with an analysis of modeling choices. First, pre-training RCM models gives significant improvements in both measuring semantic similarity and capturing human judgements (compare “p” vs. “r” results.) Second, the number of relations used for RCM training is an

important factor. Table 5 shows the effect on dev data of using various numbers of relations. While we see improvements from XL to XXL (5 times as many relations), we get worse results on XXXL, likely because this set contains the lowest quality relations in PPDB. Finally, Table 6 shows different learning rates α_{RCM} for the RCM objective.

The baseline word2vec and the joint model have nearly the same averaged running times (2,577s and 2,644s respectively), since they have same number of threads for the CBOW objective and the joint model uses additional threads for the RCM objective. The RCM models are trained with single thread for 100 epochs. When trained on the PPDB-XXL data, it spends 2,931s on average.

5 Conclusion

We have presented a new learning objective for neural language models that incorporates prior knowledge contained in resources to improve learned word embeddings. We demonstrated that the Relation Constrained Model can lead to better semantic embeddings by incorporating resources like PPDB, leading to better language modeling, semantic similarity metrics, and predicting human semantic judgements. Our implementation is based on the word2vec package and we made it available for general use ².

We believe that our techniques have implications beyond those considered in this work. We plan to explore the embeddings suitability for other semantics tasks, including the use of resources with both typed and scored relations. Additionally, we see opportunities for jointly learning embeddings across many tasks with many resources, and plan to extend our model accordingly.

Acknowledgements Yu is supported by China Scholarship Council and by NSFC 61173073.

²<https://github.com/Gorov/JointRCM>

References

- Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Frédéric Morin, and Jean-Luc Gauvain. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer.
- Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. 2007. Greedy layer-wise training of deep networks. In *Neural Information Processing Systems (NIPS)*.
- Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2012. Joint learning of words and meaning representations for open-text semantic parsing. In *International Conference on Artificial Intelligence and Statistics*, pages 127–135.
- Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *International Conference on Machine Learning (ICML)*.
- Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research (JMLR)*, 11:625–660.
- Christiane Fellbaum. 1999. *WordNet*. Wiley Online Library.
- Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.
- Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Association for Computational Linguistics (ACL)*, pages 873–882.
- Minh-Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. In *Conference on Natural Language Learning (CoNLL)*.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*.
- Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *International Conference on Machine Learning (ICML)*.
- Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. *arXiv preprint arXiv:1206.6426*.
- Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword fifth edition. Technical report, Linguistic Data Consortium.
- Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. 2005. Randomized algorithms and nlp: using locality sensitive hash function for high speed noun clustering. In *Association for Computational Linguistics (ACL)*.
- 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 *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Association for Computational Linguistics (ACL)*.
- Benjamin Van Durme and Ashwin Lall. 2010. Online generation of locality sensitive hash signatures. In *Association for Computational Linguistics (ACL)*, pages 231–235.

Optimizing Segmentation Strategies for Simultaneous Speech Translation

Yusuke Oda Graham Neubig Sakriani Sakti Tomoki Toda Satoshi Nakamura
Graduate School of Information Science
Nara Institute of Science and Technology
Takayama, Ikoma, Nara 630-0192, Japan

{oda.yusuke.on9, neubig, ssakti, tomoki, s-nakamura}@is.naist.jp

Abstract

In this paper, we propose new algorithms for learning segmentation strategies for simultaneous speech translation. In contrast to previously proposed heuristic methods, our method finds a segmentation that directly maximizes the performance of the machine translation system. We describe two methods based on greedy search and dynamic programming that search for the optimal segmentation strategy. An experimental evaluation finds that our algorithm is able to segment the input two to three times more frequently than conventional methods in terms of number of words, while maintaining the same score of automatic evaluation.¹

1 Introduction

The performance of speech translation systems has greatly improved in the past several years, and these systems are starting to find wide use in a number of applications. Simultaneous speech translation, which translates speech from the source language into the target language in real time, is one example of such an application. When translating dialogue, the length of each utterance will usually be short, so the system can simply start the translation process when it detects the end of an utterance. However, in the case of lectures, for example, there is often no obvious boundary between utterances. Thus, translation systems require a method of deciding the timing at which to start the translation process. Using estimated ends of sentences as the timing with which to start translation, in the same way as a normal text translation, is a straightforward solution to this problem (Matusov et al., 2006). However, this approach

¹The implementation is available at <http://odaemon.com/docs/codes/greedyseg.html>.

impairs the simultaneity of translation because the system needs to wait too long until the appearance of a estimated sentence boundary. For this reason, segmentation strategies, which separate the input at appropriate positions other than end of the sentence, have been studied.

A number of segmentation strategies for simultaneous speech translation have been proposed in recent years. Fügen et al. (2007) and Bangalore et al. (2012) propose using prosodic pauses in speech recognition to denote segmentation boundaries, but this method strongly depends on characteristics of the speech, such as the speed of speaking. There is also research on methods that depend on linguistic or non-linguistic heuristics over recognized text (Rangarajan Sridhar et al., 2013), and it was found that a method that predicts the location of commas or periods achieves the highest performance. Methods have also been proposed using the phrase table (Yarmohammadi et al., 2013) or the right probability (RP) of phrases (Fujita et al., 2013), which indicates whether a phrase reordering occurs or not.

However, each of the previously mentioned methods decides the segmentation on the basis of heuristics, so the impact of each segmentation strategy on translation performance is not directly considered. In addition, the mean number of words in the translation unit, which strongly affects the delay of translation, cannot be directly controlled by these methods.²

In this paper, we propose new segmentation algorithms that directly optimize translation performance given the mean number of words in the translation unit. Our approaches find appropriate segmentation boundaries incrementally using greedy search and dynamic programming. Each boundary is selected to explicitly maximize trans-

²The method using RP can decide relative frequency of segmentation by changing a parameter, but guessing the length of a translation unit from this parameter is not trivial.

lation accuracy as measured by BLEU or another evaluation measure.

We evaluate our methods on a speech translation task, and we confirm that our approaches can achieve translation units two to three times as fine-grained as other methods, while maintaining the same accuracy.

2 Optimization Framework

Our methods use the outputs of an existing machine translation system to learn a segmentation strategy. We define $\mathcal{F} = \{\mathbf{f}_j : 1 \leq j \leq N\}$, $\mathcal{E} = \{e_j : 1 \leq j \leq N\}$ as a parallel corpus of source and target language sentences used to train the segmentation strategy. N represents the number of sentences in the corpus. In this work, we consider sub-sentential segmentation, where the input is already separated into sentences, and we want to further segment these sentences into shorter units. In an actual speech translation system, these sentence boundaries can be estimated automatically using a method like the period estimation mentioned in Rangarajan Sridhar et al. (2013). We also assume the machine translation system is defined by a function $MT(\mathbf{f})$ that takes a string of source words \mathbf{f} as an argument and returns the translation result \hat{e} .³

We will introduce individual methods in the following sections, but all follow the general framework shown below:

1. Decide the mean number of words μ and the machine translation evaluation measure EV as parameters of algorithm. We can use an automatic evaluation measure such as BLEU (Papineni et al., 2002) as EV . Then, we calculate the number of sub-sentential segmentation boundaries K that we will need to insert into \mathcal{F} to achieve an average segment length μ :

$$K := \max \left(0, \left\lfloor \frac{\sum_{\mathbf{f} \in \mathcal{F}} |\mathbf{f}|}{\mu} \right\rfloor - N \right). \quad (1)$$

2. Define \mathcal{S} as a set of positions in \mathcal{F} in which we will insert segmentation boundaries. For example, if we will segment the first sentence after the third word and the third sentence after the fifth word, then $\mathcal{S} = \{\langle 1, 3 \rangle, \langle 3, 5 \rangle\}$.

³In this work, we do not use the history of the language model mentioned in Bangalore et al. (2012). Considering this information improves the MT performance and we plan to include this in our approach in future work.

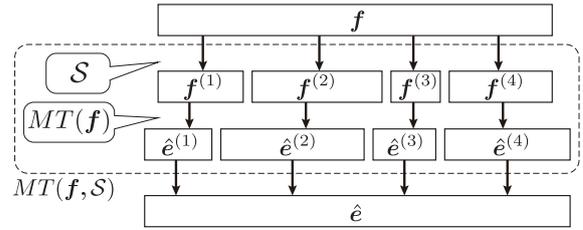


Figure 1: Concatenated translation $MT(\mathbf{f}, \mathcal{S})$.

Based on this representation, choose K segmentation boundaries in \mathcal{F} to make the set \mathcal{S}^* that maximizes an evaluation function ω as below:

$$\mathcal{S}^* := \arg \max_{\mathcal{S} \in \{\mathcal{S}' : |\mathcal{S}'| = K\}} \omega(\mathcal{S}; \mathcal{F}, \mathcal{E}, EV, MT). \quad (2)$$

In this work, we define ω as the sum of the evaluation measure for each parallel sentence pair $\langle \mathbf{f}_j, e_j \rangle$:

$$\omega(\mathcal{S}) := \sum_{j=1}^N EV(MT(\mathbf{f}_j, \mathcal{S}), e_j), \quad (3)$$

where $MT(\mathbf{f}, \mathcal{S})$ represents the concatenation of all partial translations $\{MT(\mathbf{f}^{(n)})\}$ given the segments \mathcal{S} as shown in Figure 1.

Equation (3) indicates that we assume all parallel sentences to be independent of each other, and the evaluation measure is calculated for each sentence separately. This locality assumption eases efficient implementation of our algorithm, and can be realized using a sentence-level evaluation measure such as BLEU+1 (Lin and Och, 2004).

3. Make a segmentation model $M_{\mathcal{S}^*}$ by treating the obtained segmentation boundaries \mathcal{S}^* as positive labels, all other positions as negative labels, and training a classifier to distinguish between them. This classifier is used to detect segmentation boundaries at test time.

Steps 1. and 3. of the above procedure are trivial. In contrast, choosing a good segmentation according to Equation (2) is difficult and the focus of the rest of this paper. In order to exactly solve Equation (2), we must perform brute-force search over all possible segmentations unless we make some assumptions about the relation between the ω yielded by different segmentations. However, the number of possible segmentations is exponentially large, so brute-force search is obviously intractable. In the following sections, we propose 2

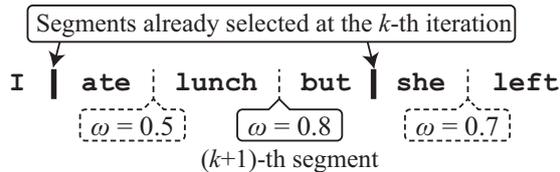


Figure 2: Example of greedy search.

Algorithm 1 Greedy segmentation search

```

 $\mathcal{S}^* \leftarrow \emptyset$ 
for  $k = 1$  to  $K$  do
     $\mathcal{S}^* \leftarrow \mathcal{S}^* \cup \left\{ \arg \max_{s \notin \mathcal{S}^*} \omega(\mathcal{S}^* \cup \{s\}) \right\}$ 
end for
return  $\mathcal{S}^*$ 

```

methods that approximately search for a solution to Equation (2).

2.1 Greedy Search

Our first approximation is a greedy algorithm that selects segmentation boundaries one-by-one. In this method, k already-selected boundaries are left unchanged when deciding the $(k + 1)$ -th boundary. We find the unselected boundary that maximizes ω and add it to \mathcal{S} :

$$\mathcal{S}_{k+1} = \mathcal{S}_k \cup \left\{ \arg \max_{s \notin \mathcal{S}_k} \omega(\mathcal{S}_k \cup \{s\}) \right\}. \quad (4)$$

Figure 2 shows an example of this process for a single sentence, and Algorithm 1 shows the algorithm for calculating K boundaries.

2.2 Greedy Search with Feature Grouping and Dynamic Programming

The method described in the previous section finds segments that achieve high translation performance for the training data. However, because the translation system MT and evaluation measure EV are both complex, the evaluation function ω includes a certain amount of noise. As a result, the greedy algorithm that uses only ω may find a segmentation that achieves high translation performance in the training data by chance. However, these segmentations will not generalize, reducing the performance for other data.

We can assume that this problem can be solved by selecting more consistent segmentations of the training data. To achieve this, we introduce a constraint that all positions that have similar characteristics must be selected at the same time. Specifically, we first group all positions in the source

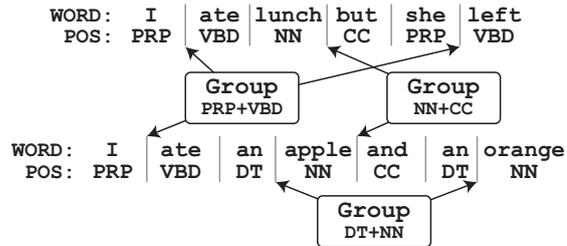


Figure 3: Grouping segments by POS bigrams.

sentences using features of the position, and introduce a constraint that all positions with identical features must be selected at the same time. Figure 3 shows an example of how this grouping works when we use the POS bigram surrounding each potential boundary as our feature set.

By introducing this constraint, we can expect that features which have good performance overall will be selected, while features that have relatively bad performance will not be selected even if good performance is obtained when segmenting at a specific location. In addition, because all positions can be classified as either segmented or not by evaluating whether the corresponding feature is in the learned feature set or not, it is not necessary to train an additional classifier for the segmentation model when using this algorithm. In other words, this constraint conducts a kind of feature selection for greedy search.

In contrast to Algorithm 1, which only selected one segmentation boundary at once, in our new setting there are multiple positions selected at one time. Thus, we need to update our search algorithm to handle this setting. To do so, we use dynamic programming (DP) together with greedy search. Algorithm 2 shows our *Greedy+DP* search algorithm. Here, $c(\phi; \mathcal{F})$ represents the number of appearances of ϕ in the set of source sentences \mathcal{F} , and $\mathcal{S}(\mathcal{F}, \Phi)$ represents the set of segments defined by both \mathcal{F} and the set of features Φ .

The outer loop of the algorithm, like *Greedy*, iterates over all \mathcal{S} of size 1 to K . The inner loop examines all features that appear exactly j times in \mathcal{F} , and measures the effect of adding them to the best segmentation with $(k - j)$ boundaries.

2.3 Regularization by Feature Count

Even after we apply grouping by features, it is likely that noise will still remain in the less frequently-seen features. To avoid this problem, we introduce regularization into the *Greedy+DP* algorithm, with the evaluation function ω rewrites

Algorithm 2 *Greedy+DP* segmentation search

$\Phi_0 \leftarrow \emptyset$
for $k = 1$ **to** K **do**
 for $j = 0$ **to** $k - 1$ **do**
 $\Phi' \leftarrow \{\phi : c(\phi; \mathcal{F}) = k - j \wedge \phi \notin \Phi_j\}$
 $\Phi_{k,j} \leftarrow \Phi_j \cup \left\{ \arg \max_{\phi \in \Phi'} \omega(\mathcal{S}(\mathcal{F}, \Phi_j \cup \{\phi\})) \right\}$
 end for
 $\Phi_k \leftarrow \arg \max_{\Phi \in \{\Phi_{k,j} : 0 \leq j < k\}} \omega(\mathcal{S}(\mathcal{F}, \Phi))$
end for
return $\mathcal{S}(\mathcal{F}, \Phi_K)$

ten as below:

$$\omega_\alpha(\Phi) := \omega(\mathcal{S}(\mathcal{F}, \Phi)) - \alpha|\Phi|. \quad (5)$$

The coefficient α is the strength of the regularization with regards to the number of selected features. A larger α will result in a larger penalty against adding new features into the model. As a result, the *Greedy+DP* algorithm will value frequently appearing features. Note that the method described in the previous section is equal to the case of $\alpha = 0$ in this section.

2.4 Implementation Details

Our *Greedy* and *Greedy+DP* search algorithms are completely described in Algorithms 1 and 2. However, these algorithms require a large amount of computation and simple implementations of them are too slow to finish in realistic time. Because the heaviest parts of the algorithm are the calculation of *MT* and *EV*, we can greatly improve efficiency by memoizing the results of these functions, only recalculating on new input.

3 Experiments

3.1 Experimental Settings

We evaluated the performance of our segmentation strategies by applying them to English-German and English-Japanese TED speech translation data from WIT3 (Cettolo et al., 2012). For English-German, we used the TED data and splits from the IWSLT2013 evaluation campaign (Cettolo et al., 2013), as well as 1M sentences selected from the out-of-domain training data using the method of Duh et al. (2013). For English-Japanese, we used TED data and the dictionary entries and sentences from EIJIRO.⁴ Table 1 shows summaries of the datasets we used.

⁴<http://eowp.alc.co.jp/info2/>

<i>f-e</i>	Type	#words	
		<i>f</i>	<i>e</i>
En-De	Train <i>MT</i>	21.8M	20.3M
	Train Seg.	424k	390k
	Test	27.6k	25.4k
En-Ja	Train <i>MT</i>	13.7M	19.7M
	Train Seg.	401k	550k
	Test	8.20k	11.9k

Table 1: Size of *MT* training, segmentation training and testing datasets.

We use the Stanford POS Tagger (Toutanova et al., 2003) to tokenize and POS tag English and German sentences, and KyTea (Neubig et al., 2011) to tokenize Japanese sentences. A phrase-based machine translation (PBMT) system learned by Moses (Koehn et al., 2007) is used as the translation system *MT*. We use BLEU+1 as the evaluation measure *EV* in the proposed method. The results on the test data are evaluated by BLEU and RIBES (Isozaki et al., 2010), which is an evaluation measure more sensitive to global reordering than BLEU.

We evaluated our algorithm and two conventional methods listed below:

Greedy is our first method that uses simple greedy search and a linear SVM (using surrounding word/POS 1, 2 and 3-grams as features) to learn the segmentation model.

Greedy+DP is the algorithm that introduces grouping the positions in the source sentence by POS bigrams.

Punct-Predict is the method using predicted positions of punctuation (Rangarajan Sridhar et al., 2013).

RP is the method using right probability (Fujita et al., 2013).

3.2 Results and Discussion

Figures 4 and 5 show the results of evaluation for each segmentation strategy measured by BLEU and RIBES respectively. The horizontal axis is the mean number of words in the generated translation units. This value is proportional to the delay experienced during simultaneous speech translation (Rangarajan Sridhar et al., 2013) and thus a smaller value is desirable.

RP, *Greedy*, and *Greedy+DP* methods have multiple results in these graphs because these methods have a parameter that controls segmentation frequency. We move this parameter from no segmentation (sentence-based translation) to

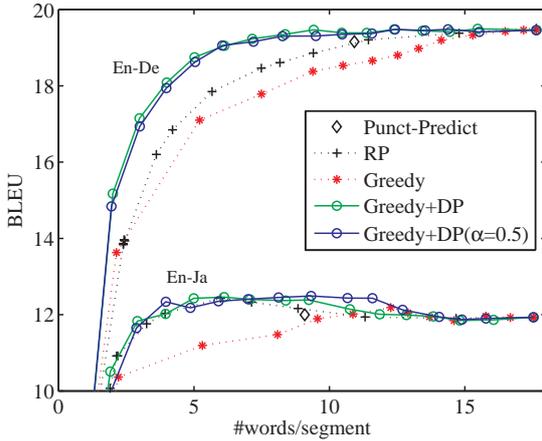


Figure 4: BLEU score of test set.

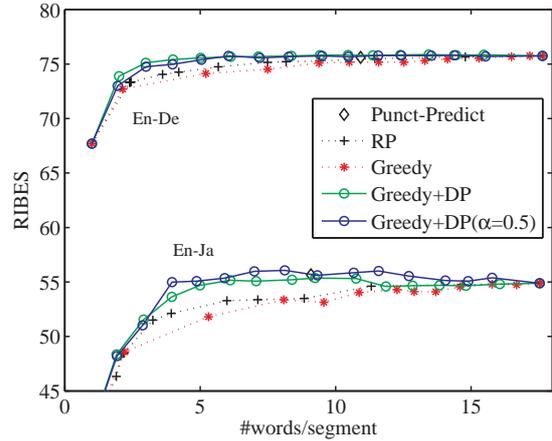


Figure 5: RIBES score of test set.

segmenting every possible boundary (word-based translation) and evaluate the results.

First, focusing on the *Greedy* method, we can see that it underperforms the other methods. This is a result of over-fitting as will be described in detail later. In contrast, the proposed *Greedy+DP* method shows high performance compared to the other methods. Especially, the result of BLEU on the English-German and the RIBES on both language pairs show higher performance than *RP* at all speed settings. *Punct-Predict* does not have an adjustable parameter, so we can only show one point. We can see that *Greedy+DP* can begin translation about two to three times faster than *Punct-Predict* while maintaining the same performance.

Figure 6 shows the BLEU on the training data. From this figure, it is clear that *Greedy* achieves much higher performance than *Greedy+DP*. From this result, we can see that the *Greedy* algorithm is choosing a segmentation that achieves high accuracy on the training data but does not generalize to the test data. In contrast, the grouping constraint in the *Greedy+DP* algorithm is effectively suppressing this overfitting.

The mean number of words μ can be decided independently from other information, but a configuration of μ affects tradeoff relation between translation accuracy and simultaneity. For example, smaller μ makes faster translation speed but it also makes less translation accuracy. Basically, we should choose μ by considering this tradeoff.

4 Conclusion and Future Work

We proposed new algorithms for learning a segmentation strategy in simultaneous speech trans-

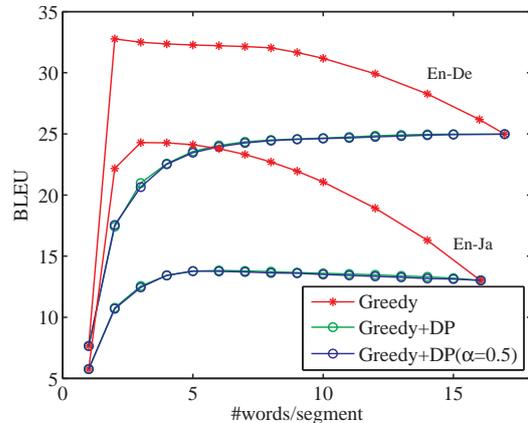


Figure 6: BLEU score of training set.

lation. Our algorithms directly optimize the performance of a machine translation system according to an evaluation measure, and are calculated by greedy search and dynamic programming. Experiments show our *Greedy+DP* method effectively separates the source sentence into smaller units while maintaining translation performance.

With regards to future work, it has been noted that translation performance can be improved by considering the previously translated segment when calculating LM probabilities (Rangarajan Sridhar et al., 2013). We would like to expand our method to this framework, although incorporation of context-sensitive translations is not trivial. In addition, the *Greedy+DP* algorithm uses only one feature per a position in this paper. Using a variety of features is also possible, so we plan to examine expansions of our algorithm to multiple overlapping features in future work.

Acknowledgements

Part of this work was supported by JSPS KAKENHI Grant Number 24240032.

References

- Srinivas Bangalore, Vivek Kumar Rangarajan Sridhar, Prakash Kolan, Ladan Golipour, and Aura Jimenez. 2012. Real-time incremental speech-to-speech translation of dialogs. In *Proc. NAACL HLT*, pages 437–445.
- Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit³: Web inventory of transcribed and translated talks. In *Proc. EAMT*, pages 261–268.
- Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2013. Report on the 10th iwslt evaluation campaign. In *Proc. IWSLT*.
- Kevin Duh, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada. 2013. Adaptation data selection using neural language models: Experiments in machine translation. In *Proc. ACL*, pages 678–683.
- Christian Fügen, Alex Waibel, and Muntsin Kolss. 2007. Simultaneous translation of lectures and speeches. *Machine Translation*, 21(4):209–252.
- Tomoki Fujita, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2013. Simple, lexicalized choice of translation timing for simultaneous speech translation. In *InterSpeech*.
- Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *Proc. EMNLP*, pages 944–952.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. ACL*, pages 177–180.
- Chin-Yew Lin and Franz Josef Och. 2004. Orange: A method for evaluating automatic evaluation metrics for machine translation. In *Proc. COLING*.
- Evgeny Matusov, Arne Mauser, and Hermann Ney. 2006. Automatic sentence segmentation and punctuation prediction for spoken language translation. In *Proc. IWSLT*, pages 158–165.
- Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable japanese morphological analysis. In *Proc. NAACL HLT*, pages 529–533.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proc. ACL*, pages 311–318.
- Vivek Kumar Rangarajan Sridhar, John Chen, Srinivas Bangalore, Andrej Ljolje, and Rathinavelu Chengalvarayan. 2013. Segmentation strategies for streaming speech translation. In *Proc. NAACL HLT*, pages 230–238.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. NAACL*, pages 173–180.
- Mahsa Yarmohammadi, Vivek Kumar Rangarajan Sridhar, Srinivas Bangalore, and Baskaran Sankaran. 2013. Incremental segmentation and decoding strategies for simultaneous translation. In *Proc. IJCNLP*, pages 1032–1036.

A joint inference of deep case analysis and zero subject generation for Japanese-to-English statistical machine translation

Taku Kudo, Hiroshi Ichikawa, Hideto Kazawa
Google Japan
{taku,ichikawa,kazawa}@google.com

Abstract

We present a simple joint inference of deep case analysis and zero subject generation for the pre-ordering in Japanese-to-English machine translation. The detection of subjects and objects from Japanese sentences is more difficult than that from English, while it is the key process to generate correct English word orders. In addition, subjects are often omitted in Japanese when they are inferable from the context. We propose a new Japanese deep syntactic parser that consists of pointwise probabilistic models and a global inference with linguistic constraints. We applied our new deep parser to pre-ordering in Japanese-to-English SMT system and show substantial improvements in automatic evaluations.

1 Introduction

Japanese to English translation is known to be one of the most difficult language pair for statistical machine translation (SMT). It has been widely believed for years that the difference of word orders, i.e., Japanese is an SOV language, while English is an SVO language, makes the English-to-Japanese and Japanese-to-English translation difficult. However, simple, yet powerful pre-ordering techniques have made this argument a thing of the past (Isozaki et al., 2010b; Komachi et al., 2006; Fei and Michael, 2004; Lerner and Petrov, 2013; Wu et al., 2011; Katz-Brown and Collins, 2008; Neubig et al., 2012; Hoshino et al., 2013). Pre-ordering processes the source sentence in such a way that word orders appear closer to their final positions on the target side.

While many successes of English-to-Japanese translation have been reported recently, the quality improvement of Japanese-to-English translation is still small even with the help of pre-ordering (Goto

et al., 2013). We found that there are two major issues that make Japanese-to-English translation difficult. One is that Japanese subject and object cannot easily be identified compared to English, while their detections are the key process to generate correct English word orders. Japanese surface syntactic structures are not always corresponding to their deep structures, i.e., semantic roles. The other is that Japanese is a pro-drop language in which certain classes of pronouns may be omitted when they are pragmatically inferable. In Japanese-to-English translation, these omitted pronouns have to be generated properly.

There are several researches that focused on the pre-ordering with Japanese deep syntactic analysis (Komachi et al., 2006; Hoshino et al., 2013) and zero pronoun generation (Taira et al., 2012) for Japanese-to-English translation. However, these two issues have been considered independently, while they heavily rely on one another.

In this paper, we propose a simple joint inference which handles both Japanese deep structure analysis and zero pronoun generation. To the best of our knowledge, this is the first study that addresses these two issues at the same time.

This paper is organized as follows. First, we describe why Japanese-to-English translation is difficult. Second, we show the basic idea of this work and its implementation based on pointwise probabilistic models and a global inference with an integer linear programming (ILP). Several experiments are employed to confirm that our new model can improve the Japanese to English translation quality.

2 What makes Japanese-to-English translation difficult?

Japanese syntactic relations between arguments and predicates are usually specified by particles. There are several types of particles, but we focus on *が* (*ga*), *を* (*wo*) and *は* (*wa*) for the sake of

Table 1: An example of difficult sentence for parsing

Sentence:	今日 は お酒 が 飲める。
Gloss:	today <i>wa</i> _{TOP} liquor <i>ga</i> _{NOM} can_drink.
Translation:	(I) can drink liquor today.

simplicity ¹.

- *ga* is usually a subject marker. However, it becomes an object marker if the predicate has a potential voice type, which is usually translated into *can*, *be able to*, *want to*, or *would like to*.
- *wo* is an object marker.
- *wa* is a topic case marker. The topic can be anything that a speaker wants to talk about. It can be subject, object, location, time or any other grammatical elements.

We cannot always identify Japanese subject and object only by seeing the surface case markers *ga*, *wo* and *wa*. Especially the topic case marker is problematic, since there is no concept of topic in English. It is necessary to get a deep interpretation of topic case markers in order to develop accurate Japanese-to-English SMT systems.

Another big issue is that Japanese subject (or even an object) can be omitted when they can pragmatically be inferable from the context. Such a pronoun-dropping is not a unique phenomenon in Japanese actually. For instance, Spanish also allows to omit pronouns. However, the inflectional suffix of Spanish verbs include a hint of the person of the subject. On the other hand, inferring Japanese subjects is more difficult than Spanish, since Japanese verbs usually do not have any grammatical cues to tell the subject type.

Table 1 shows an example Japanese sentence which cannot be parsed only with the surface structure. The second token *wa* specifies the relation between 今日 (*today*) and 飲める (*can drink*). Human can easily tell that the relation of them is not a subject but an adverb (time). The topic case marker *wa* implies that the time when the speaker drinks liquor is the focus of this sentence. The 4th token *ga* indicates the relation between お酒 (*liquor*) and 飲める (*can drink*). Since the predicate has a potential voice (*can drink*), the *ga* particle should be interpreted as an object here. In

¹Other case markers are less frequent than these three markers

this sentence, the subject is omitted. In general, it is unknown who speaks this sentence, but the first person is a natural interpretation in this context.

Another tricky phenomenon is that detecting voice type is not always deterministic. There are several ways to generate a potential voice in Japanese, but we usually put the suffix word れる (*reru*) or られる (*rareru*) after predicates. However, these suffix words are also used for a passive voice.

In summary, we can see that the following four factors are the potential causes that make the Japanese parsing difficult.

- Japanese voice type detection is not straightforward. *reru* or *rareru* are used either for passive or potential voice.
- surface case *ga* changes its interpretation from subject to object when the predicate has a potential voice.
- topic case marker *wa* is used as a topic case marker which doesn't exist in English. Topic is either subject, object or any grammatical elements depending on the context.
- Japanese subject is often omitted when it is inferable from the context. There is no cue to tell the subject person in verb suffix (inflection) like in Spanish verbs

We should note that they are not always independent issues. For instance, the deep case detection helps to tell the voice type, and vice versa.

Another note is that they are unique issues observed only in Japanese-to-English translation. In English-to-Japanese translation, it is acceptable to generate Japanese sentences that do not use Japanese topic markers *wa*. Also, generating Japanese pronoun from English pronoun is acceptable, although it sounds redundant and unnatural for native speakers.

3 A joint inference of deep case analysis and zero subject generation

3.1 Probabilistic model over predicate-argument structures

Our deep parser runs on the top of a dependency parse tree. First, it extracts all predicates and their arguments from a dependency tree by using manual rules over POS tags. Since our pre-ordering system generates the final word orders from a labeled dependency tree, we formalize our deep

parsing task as a simple labeling problem over dependency links, where the label indicates the deep syntactic roles between head and modifier.

We here define a joint probability over a predicate and its arguments as follows:

$$P(p, z, v, A, S, D) \quad (1)$$

where

- p : a predicate
- z : a zero subject candidate for p . $z \in Z = \{I, you, we, it, he/she, imperative, already_exists\}$
- v : voice type of the predicate p . $v \in V = \{active, passive, potential\}$
- $a_k \in A$: k -th argument which modifies or is modified by the predicate².
- $d_k \in D$: deep case label which represents a deep relation between a_k and p . $d \in \{subject, object, other\}$, where *other* means that deep case is neither subject nor object.
- $s_k \in S$: surface relation (surface case marker) between a_k and p .

We assume that a predicate p is independent from other predicates in a sentence. This assumption allows us to estimate the deep structures of p separately, with no regard to which decisions are made in other predicates.

An optimal zero subject label z , deep cases D , and voice type v for a given predicate p can be solved as the following optimization problem.

$$\langle \hat{z}, \hat{v}, \hat{D} \rangle = \underset{z, v, D}{argmax} P(p, z, v, A, S, D)$$

Since the inference of this joint probability is difficult, we decompose $P(p, z, v, A, S, D)$ into small independent sub models:

$$P(p, z, v, A, S, D) \approx P_z(z|p, A, S)P_v(v|p, A, S)P_d(D|p, v, A, S)P(p, A, S) \quad (2)$$

We do not take the last term $P(p, A, S)$ into consideration, since it is constant for the optimization. In the next sections, we describe how these probabilities P_z , P_d , and P_v are computed.

²Generally, an argument modifies a predicate, but in relative clauses, a predicate modifies an argument

3.1.1 Zero subject model: $P_z(z|p, A, S)$

This model estimates the syntactic zero subject³ of the predicate p . For instance, $z=I$ means that the subject of p is omitted and its type is first person. $z=imperative$ means that we do not need to augment a subject because the predicate is imperative. $z=already_exists$ means that a subject already appears in the sentence. A maximum entropy classifier is used in our zero subject model, which takes the contextual features extracted from p , A , and S .

3.1.2 Voice type model: $P_v(v|p, A, S)$

This model estimates the voice type of a predicate. We also use a maximum entropy classifier for this model. This classifier is used only when the predicate has the ambiguous suffix *reru* or *rareru*. If the predicate does not have any ambiguous suffix, this model returns pre-defined voice types with very high probabilities.

3.1.3 Deep case model: $P_d(D|p, v, A, S)$

This model estimates the deep syntactic role between a predicate p and its arguments A . This model helps to resolve the deep cases when their surface cases are topic. We define P_d as follows after introducing an independent assumption over predicate-argument structures:

$$P(D|p, v, A, S) \approx \prod_i [\max(p(d_i|a_i, p) - m(s_i, d_i, v), \delta)].$$

$p(d|a, p)$ models the deep relation between p and a . We use a maximum likelihood estimation for $p(d|a, p)$:

$$p(d = subj|a, p) = \frac{freq(s = ga, a, \text{active form of } p)}{freq(a, \text{active form of } p)}$$

$$p(d = obj|a, p) = \frac{freq(s = wo, a, \text{active form of } p)}{freq(a, \text{active form of } p)},$$

where $freq(s = ga, a, \text{active form of } p)$ is the frequency of how often an argument a and p appears with the surface case ga . The frequencies are aggregated only when the predicate appear in active voice. If the voice type is active, we can safely assume that the surface cases ga and wo correspond to subject and object respectively. We compute the frequencies from a large amount of auto-parsed data.

$m(s, d, v)$ is a non-negative penalty variable describing how the deep case d generates the surface case s depending on the voice type v . Since

³Here *syntactic subject* means the subject which takes the voice type into account.

the number of possible surface cases, deep cases, and voice types are small, we define this penalty manually by referring to the Japanese grammar book (descriptive grammar research group, 2009). We use these manually defined penalties in order to put more importance on syntactic preferences rather than those of semantics. Even if a predicate-augment structure is semantically irrelevant, we take this structure as long as it is syntactically correct in order to avoid SMT from generating liberal translations.

δ is a very small positive constant to avoid zero probability.

3.2 Joint inference with linguistic constraints

Our initial model (2) assumes that zero subjects and deep cases are generated independently. However, this assumption does not always capture real linguistic phenomena. English is a subject-prominent language in which almost all sentences (or predicates) must have a subject. This implies that it is more reasonable to introduce strong linguistic constraints to the final solution for pre-ordering, which are described as follows:

- Subject is a mandatory role. A subject must be inferred either by zero subject or deep case model ⁴. When the voice type is passive, an object role in D is considered as a syntactic subject.
- A predicate can not have multiple subjects and objects respectively.

These two constraints avoid the model from inferring syntactically irrelevant solutions.

In order to find the result with the constraints above, we formalize our model as an integer linear programming, ILP. Let $\{x_1, \dots, x_n\}$ be binary variables, i.e., $x_i \in \{0, 1\}$. x_i corresponds to the binary decisions in our model, e.g., $x_k = 1$ if $d_i = \text{subj}$ and $v = \text{active}$. Let $\{p_1, \dots, p_n\}$ be probability vector corresponding to the binary decisions. ILP can be formalized as a mathematical problem, in which the objective function and the constraints are linear:

$$\{\hat{x}_1, \dots, \hat{x}_n\} = \underset{\{x_1, \dots, x_n\} \in \{0, 1\}^n}{\operatorname{argmax}} \sum_{i=1}^n \log(p_i) x_i$$

s.t. linear constraints over $\{x_1, \dots, x_n\}$.

After taking the log of (2), our optimization model can be converted into an ILP. Also, the constraints

⁴*imperative* is also handled as an invisible subject

described above can be represented as linear equations over binary variables X . We leave the details of the representations to (Punyakanok et al., 2004; Iida and Poesio, 2011).

3.3 Japanese pre-ordering with deep parser

We use a simple rule-based approach to make pre-ordered Japanese sentences from our deep parse trees, which is similar to the algorithms described in (Komachi et al., 2006; Katz-Brown and Collins, 2008; Hoshino et al., 2013). First, we naively reverse all the *bunsetsu*-chunks ⁵. Then, we move a subject chunk just before its predicate. This process converts SOV to SVO. When the subject is omitted, we generate a subject with our deep parser and insert it to a subject position in the source sentence. There are three different ways to generate a subject.

1. Generate real Japanese words (Insert 私 (*I*), あなた (*you*).. etc)
2. Generate virtual seed Japanese words (Insert *1st_person*, *2nd_person*..., which are not in the Japanese lexicon.)
3. Generate only a single virtual seed Japanese word regardless of the subject type. (Insert *zero_subject*)

1) is the most aggressive method, but it causes completely incorrect translations if the detection of subject type fails. 2) and 3) is rather conservative, since they leave SMT to generate English pronouns.

We decided to use the following hybrid approach, since it shows the best performance in our preliminary experiments.

- In the training of SMT, use 3).
- In decoding, use 1) if the input sentence only has one predicate. Otherwise, use 3).

3.4 Examples of parsing results

Table 2 shows examples of our deep parser output. It can be seen that our parser can correctly identify the deep case of topic case markers *wa*.

⁵*bunsetsu* is a basic Japanese grammatical unit consisting of one content word and functional words.

Table 2: Examples of deep parser output

今日は (<i>today wa</i>) _{d=other}	酒が (<i>liquor ga</i>) _{d=obj}	飲める (<i>can drink</i>) _{v=potential, z=I}
ニュースが (<i>news ga</i>) _{d=subj}	伝えられた (<i>was broadcast</i>) _{v=passive, z=already_exist}	
パスタは (<i>pasta wa</i>) _{d=obj}	食べましたか (<i>ate+question</i>) _{v=active, z=you}	
あなたは (<i>you wa</i>) _{d=subj}	食べましたか (<i>ate+question</i>) _{v=active, z=already_exist}	

4 Experiments

4.1 Experimental settings

We carried out all our experiments using a state-of-the-art phrase-based statistical Japanese-to-English machine translation system (Och, 2003) with pre-ordering. During the decoding, we use the reordering window (distortion limit) to 4 words. For parallel training data, we use an in-house collection of parallel sentences. These come from various sources with a substantial portion coming from the web. We trained our system on about 300M source words. Our test set contains about 10,000 sentences randomly sampled from the web.

The dependency parser we apply is an implementation of a shift-reduce dependency parser which uses a *bunsetsu*-chunk as a basic unit for parsing (Kudo and Matsumoto, 2002).

The zero subject and voice type models were trained with about 20,000 and 5,000 manually annotated web sentences respectively. In order to simplify the rating tasks for our annotators, we extracted only one candidate predicate from a sentence for annotations.

We tested the following six systems.

- **baseline**: no pre-ordering.
- **surface reordering** : pre-ordering only with surface dependency relations.
- **independent deep reordering**: pre-ordering using deep parser without global linguistic constraints.
- **independent deep reordering + zero subject**: pre-ordering using deep parser and zero subject generation without global linguistic constraints.
- **joint deep reordering**: pre-ordering using our new deep parser with global linguistic constraints.
- **joint deep reordering + zero-subject**: pre-ordering using deep parser and zero subject generation with global linguistic constraints.

Table 3: Results for different reordering methods

System	BLEU	RIBES
baseline (no reordering)	16.15	52.67
surface reordering	19.39	60.30
independent deep reordering	19.68	61.27
independent deep reordering + zero subj.	19.81	61.67
joint deep reordering	19.76	61.43
joint deep reordering + zero subj.	19.90	61.89

As translation metrics, we used BLEU (Papineni et al., 2002), as well as RIBES (Isozaki et al., 2010a), which is designed for measuring the quality of distant language pairs in terms of word orders.

4.2 Results

Table 3 shows the experimental results for six pre-reordering systems. It can be seen that the proposed method with deep parser outperforms baseline and naive reordering with surface syntactic trees. The zero subject generation can also improve both BLEU and RIBES scores, but the improvements are smaller than those with reordering. Also, joint inference with global linguistic constraints outperforms the model which solves deep syntactic analysis and zero subject generation independently.

5 Conclusions

In this paper, we proposed a simple joint inference of deep case analysis and zero subject generation for Japanese-to-English SMT. Our parser consists of pointwise probabilistic models and a global inference with linguistic constraints. We applied our new deep parser to pre-ordering in Japanese-to-English SMT system and showed substantial improvements in automatic evaluations.

Our future work is to enhance our deep parser so that it can handle other linguistic phenomena, including causative voice, coordinations, and object ellipsis. Also, the current system is built on the top of a dependency parser. The final output of our deep parser is highly influenced by the parsing errors. It would be interesting to develop a full joint inference of dependency parsing and deep syntactic analysis.

References

- Japan descriptive grammar research group. 2009. *Contemporary Japanese grammar book 2. Part 3. Case and Syntax, Part 4. Voice*. Kuroshio Publishers.
- Xia Fei and McCord Michael. 2004. Improving a statistical mt system with automatically learned rewrite patterns. In *Proc. of ACL*.
- Isao Goto, Bin Lu, Ka Po Chow, Eiichiro Sumita, and Benjamin K Tsou. 2013. Overview of the patent machine translation task at the ntcir-10 workshop. In *Proc. of NTCIR*.
- Sho Hoshino, Yusuke Miyao, Katsuhito Sudoh, and Masaaki Nagata. 2013. Two-stage pre-ordering for japanese-to-english statistical machine translation. In *Proc. IJCNLP*.
- Ryu Iida and Massimo Poesio. 2011. A cross-lingual ilp solution to zero anaphora resolution. In *Proc. of ACL*.
- Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010a. Automatic evaluation of translation quality for distant language pairs. In *Proc. of EMNLP*. Association for Computational Linguistics.
- Hideki Isozaki, Katsuhito Sudoh, Hajime Tsukada, and Kevin Duh. 2010b. Head finalization: A simple reordering rule for sov languages. In *Proc. of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*.
- Jason Katz-Brown and Michael Collins. 2008. Syntactic reordering in preprocessing for japanese \rightarrow english translation: Mit system description for ntcir-7 patent translation task. In *Proc. of the NTCIR-7 Workshop Meeting*.
- Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2006. Phrase reordering for statistical machine translation based on predicate-argument structure. In *Proc. of the International Workshop on Spoken Language Translation*.
- Taku Kudo and Yuji Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *Proc. of CoNLL*.
- Uri Lerner and Slav Petrov. 2013. Source-side classifier preordering for machine translation. In *Proc. of EMNLP*.
- Graham Neubig, Taro Watanabe, and Shinsuke Mori. 2012. Inducing a discriminative parser to optimize machine translation reordering. In *Proc. of EMNLP*.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL*.
- Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proc. of ACL*.
- Hiroto Taira, Katsuhito Sudoh, and Masaaki Nagata. 2012. Zero pronoun resolution can improve the quality of je translation. In *Proc. of Workshop on Syntax, Semantics and Structure in Statistical Translation*.
- Xianchao Wu, Katsuhito Sudoh, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. 2011. Extracting pre-ordering rules from predicate-argument structures. In *Proc. of IJCNLP*.

A Hybrid Approach to Skeleton-based Translation

Tong Xiao^{†‡}, Jingbo Zhu^{†‡}, Chunliang Zhang^{†‡}

[†] Northeastern University, Shenyang 110819, China

[‡] Hangzhou YaTuo Company, 358 Wener Rd., Hangzhou 310012, China

{xiaotong, zhujingbo, zhangcl}@mail.neu.edu.cn

Abstract

In this paper we explicitly consider sentence skeleton information for Machine Translation (MT). The basic idea is that we translate the key elements of the input sentence using a skeleton translation model, and then cover the remain segments using a full translation model. We apply our approach to a state-of-the-art phrase-based system and demonstrate very promising BLEU improvements and TER reductions on the NIST Chinese-English MT evaluation data.

1 Introduction

Current Statistical Machine Translation (SMT) approaches model the translation problem as a process of generating a derivation of atomic translation units, assuming that every unit is drawn out of the same model. The simplest of these is the phrase-based approach (Och et al., 1999; Koehn et al., 2003) which employs a global model to process any sub-strings of the input sentence. In this way, all we need is to increasingly translate a sequence of source words each time until the entire sentence is covered. Despite good results in many tasks, such a method ignores the roles of each source word and is somewhat different from the way used by translators. For example, an important-first strategy is generally adopted in human translation - we translate the key elements/structures (or skeleton) of the sentence first, and then translate the remaining parts. This especially makes sense for some languages, such as Chinese, where complex structures are usually involved.

Note that the source-language structural information has been intensively investigated in recent studies of syntactic translation models. Some of them developed syntax-based models on complete

syntactic trees with Treebank annotations (Liu et al., 2006; Huang et al., 2006; Zhang et al., 2008), and others used source-language syntax as soft constraints (Marton and Resnik, 2008; Chiang, 2010). However, these approaches suffer from the same problem as the phrase-based counterpart and use the single global model to handle different translation units, no matter they are from the skeleton of the input tree/sentence or other not-so-important sub-structures.

In this paper we instead explicitly model the translation problem with sentence skeleton information. In particular,

- We develop a skeleton-based model which divides translation into two sub-models: a skeleton translation model (i.e., translating the key elements) and a full translation model (i.e., translating the remaining source words and generating the complete translation).
- We develop a skeletal language model to describe the possibility of translation skeleton and handle some of the long-distance word dependencies.
- We apply the proposed model to Chinese-English phrase-based MT and demonstrate promising BLEU improvements and TER reductions on the NIST evaluation data.

2 A Skeleton-based Approach to MT

2.1 Skeleton Identification

The first issue that arises is how to identify the skeleton for a given source sentence. Many ways are available. E.g., we can start with a full syntactic tree and transform it into a simpler form (e.g., removing a sub-tree). Here we choose a simple and straightforward method: a skeleton is obtained by dropping all unimportant words in the original sentence, while preserving the grammaticality. See the following for an example skeleton of a Chinese sentence.

Original Sentence (subscripts represent indices):

每_[1] 吨_[2] 海水淡化_[3] 处理_[4] 的_[5]
per ton seawater desalination treatment of

成本_[6] 在_[7] 5_[8] 元_[9] 的_[10] 基础_[11] 上_[12]
the cost 5 yuan of from

进一步_[13] 下降_[14] 。_[15]
has been further reduced .

(The cost of seawater desalination treatment has been further reduced from 5 yuan per ton.)

Sentence Skeleton (subscripts represent indices):

成本_[6] 进一步_[13] 下降_[14] 。_[15]
the cost has been further reduced .

(The cost has been further reduced.)

Obviously the skeleton used in this work can be viewed as a simplified sentence. Thus the problem is in principle the same as sentence simplification/compression. The motivations of defining the problem in this way are two-fold. First, as the skeleton is a well-formed (but simple) sentence, all current MT approaches are applicable to the skeleton translation problem. Second, obtaining simplified sentences by word deletion is a well-studied issue (Knight and Marcu, 2000; Clarke and Lapata, 2006; Galley and McKeown, 2007; Cohn and Lapata, 2008; Yamangil and Shieber, 2010; Yoshikawa et al., 2012). Many good sentence simplication/compression methods are available to our work. Due to the lack of space, we do not go deep into this problem. In Section 3.1 we describe the corpus and system employed for automatic generation of sentence skeletons.

2.2 Base Model

Next we describe our approach to integrating skeleton information into MT models. We start with an assumption that the 1-best skeleton is provided by the skeleton identification system. Then we define skeleton-based translation as a task of searching for the best target string \hat{t} given the source string and its skeleton τ :

$$\hat{t} = \arg \max_t P(t|\tau, s) \quad (1)$$

As is standard in SMT, we further assume that 1) the translation process can be decomposed into a derivation of phrase-pairs (for phrase-based models) or translation rules (for syntax-based models); 2) and a linear function $g(\cdot)$ is used to assign a model score to each derivation. Let $d_{s,\tau,t}$ (or d for short) denote a translation derivation. The

above problem can be redefined in a Viterbi fashion - we find the derivation \hat{d} with the highest model score given s and τ :

$$\hat{d} = \arg \max_d g(d) \quad (2)$$

In this way, the MT output can be regarded as the target-string encoded in \hat{d} .

To compute $g(d)$, we use a linear combination of a skeleton translation model $g_{skel}(d)$ and a full translation model $g_{full}(d)$:

$$g(d) = g_{skel}(d) + g_{full}(d) \quad (3)$$

where the skeleton translation model handles the translation of the sentence skeleton, while the full translation model is the baseline model and handles the original problem of translating the whole sentence. The motivation here is straightforward: we use an additional score $g_{skel}(d)$ to model the problem of skeleton translation and interpolate it with the baseline model. See Figure 1 for an example of applying the above model to phrase-based MT. In the figure, each source phrase is translated into a target phrase, which is represented by linked rectangles. The skeleton translation model focuses on the translation of the sentence skeleton, i.e., the solid (red) rectangles; while the full translation model computes the model score for all those phrase-pairs, i.e., all solid and dashed rectangles.

Another note on the model. Eq. (3) provides a very flexible way for model selection. While we will restrict ourself to phrase-based translation in the following description and experiments, we can choose different models/features for $g_{skel}(d)$ and $g_{full}(d)$. E.g., one may introduce syntactic features into $g_{skel}(d)$ due to their good ability in capturing structural information; and employ a standard phrase-based model for $g_{full}(d)$ in which not all segments of the sentence need to respect syntactic constraints.

2.3 Model Score Computation

In this work both the skeleton translation model $g_{skel}(d)$ and full translation model $g_{full}(d)$ resemble the usual forms used in phrase-based MT, i.e., the model score is computed by a linear combination of a group of phrase-based features and language models. In phrase-based MT, the translation problem is modeled by a derivation of phrase-pairs. Given a translation model m , a language model lm and a vector of feature weights \mathbf{w} , the model score of a derivation d is computed by

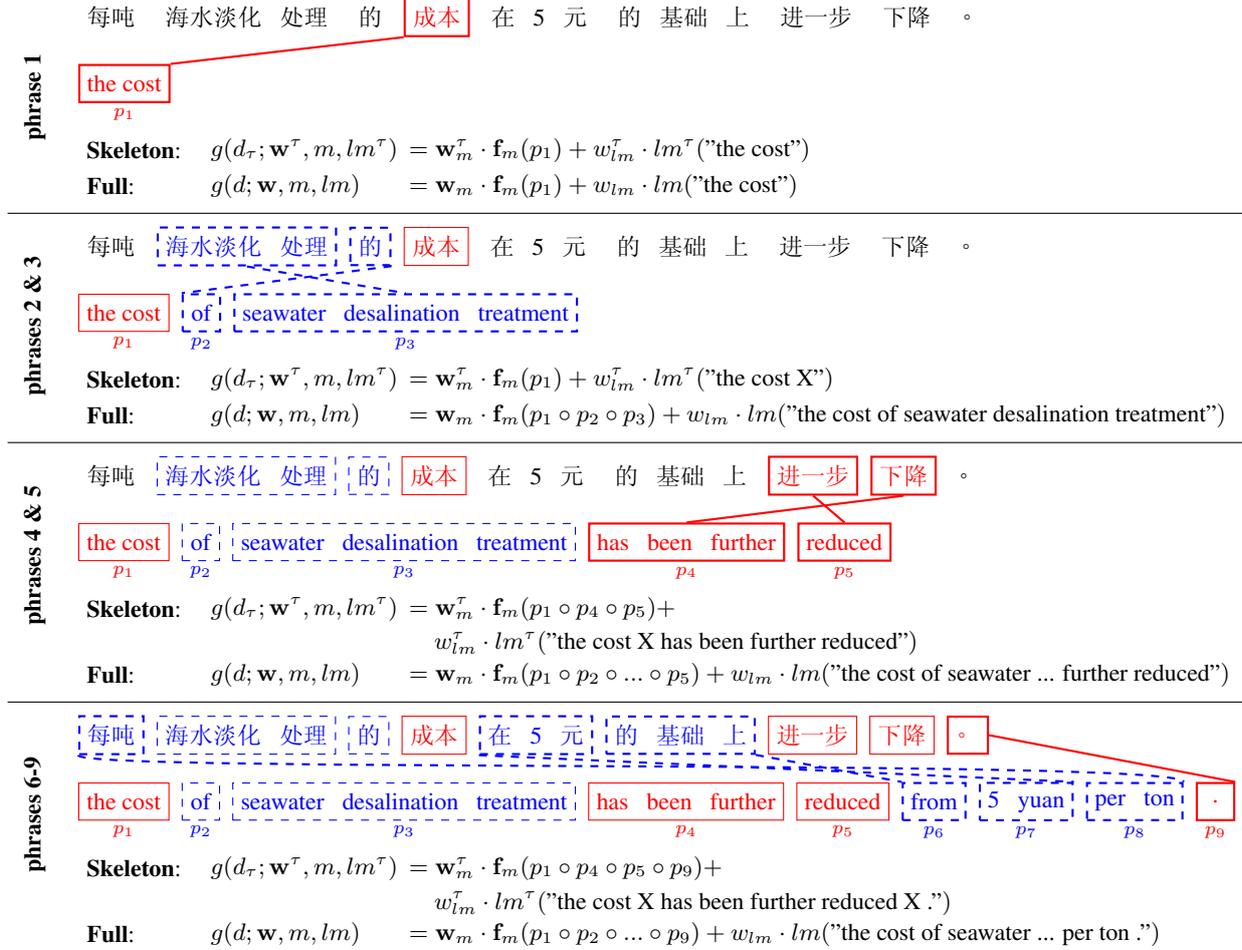


Figure 1: Example derivation and model scores for a sentence in LDC2006E38. The solid (red) rectangles represent the sentence skeleton, and the dashed (blue) rectangles represent the non-skeleton segments. X represents a slot in the translation skeleton. \circ represents composition of phrase-pairs.

$$g(d; \mathbf{w}, m, lm) = \mathbf{w}_m \cdot \mathbf{f}_m(d) + w_{lm} \cdot lm(d) \quad (4)$$

$$g_{skel}(d) \triangleq g(d; \mathbf{w}^\tau, m, lm^\tau) \quad (5)$$

$$g_{full}(d) \triangleq g(d; \mathbf{w}, m, lm) \quad (6)$$

where $\mathbf{f}_m(d)$ is a vector of feature values defined on d , and \mathbf{w}_m is the corresponding weight vector. $lm(d)$ and w_{lm} are the score and weight of the language model, respectively.

To ease modeling, we only consider *skeleton-consistent* derivations in this work. A derivation d is skeleton-consistent if no phrases in d cross skeleton boundaries (e.g., a phrase where two of the source words are in the skeleton and one is outside). Obviously, from any skeleton-consistent derivation d we can extract a skeleton derivation d_τ which covers the sentence skeleton exactly. For example, in Figure 1, the derivation of phrase-pairs $\{p_1, p_2, \dots, p_9\}$ is skeleton-consistent, and the skeleton derivation is formed by $\{p_1, p_4, p_5, p_9\}$.

Then, we can simply define $g_{skel}(d)$ and $g_{full}(d)$ as the model scores of d_τ and d :

This model makes the skeleton translation and full translation much simpler because they perform in the same way of string translation in phrase-based MT. Both $g_{skel}(d)$ and $g_{full}(d)$ share the same translation model m which can easily learned from the bilingual data¹. On the other hand, it has different feature weight vectors for individual models (i.e., \mathbf{w} and \mathbf{w}^τ).

For language modeling, lm is the standard n -gram language model adopted in the baseline system. lm^τ is a skeletal language for estimating the well-formedness of the translation skeleton. Here a translation skeleton is a target string where all segments of non-skeleton translation are generalized to a symbol X. E.g., in Figure 1, the trans-

¹In $g_{skel}(d)$, we compute the reordering model score on the skeleton though it is learned from the full sentences. In this way the reordering problems in skeleton translation and full translation are distinguished and handled separately.

lation skeleton is ‘*the cost X has been further reduced X*.’, where two Xs represent non-skeleton segments in the translation. In such a way of string representation, the skeletal language model can be implemented as a standard n -gram language model, that is, a string probability is calculated by a product of a sequence of n -gram probabilities (involving normal words and X). To learn the skeletal language model, we replace non-skeleton parts of the target sentences in the bilingual corpus to Xs using the source sentence skeletons and word alignments. The skeletal language model is then trained on these generalized strings in a standard way of n -gram language modeling.

By substituting Eq. (4) into Eqs. (5) and (6), and then Eqs. (3) and (2), we have the final model used in this work:

$$\hat{d} = \arg \max_d \left(\mathbf{w}_m \cdot \mathbf{f}_m(d) + w_{lm} \cdot lm(d) + \mathbf{w}_m^\tau \cdot \mathbf{f}_m(d_\tau) + w_{lm}^\tau \cdot lm^\tau(d_\tau) \right) \quad (7)$$

Figure 1 shows the translation process and associated model scores for the example sentence. Note that this method does not require any new translation models for implementation. Given a baseline phrase-based system, all we need is to learn the feature weights \mathbf{w} and \mathbf{w}^τ on the development set (with source-language skeleton annotation) and the skeletal language model lm^τ on the target-language side of the bilingual corpus. To implement Eq. (7), we can perform standard decoding while “doubly weighting” the phrases which cover a skeletal section of the sentence, and combining the two language models and the translation model in a linear fashion.

3 Evaluation

3.1 Experimental Setup

We experimented with our approach on Chinese-English translation using the NiuTrans open-source MT toolkit (Xiao et al., 2012). Our bilingual corpus consists of 2.7M sentence pairs. All these sentences were aligned in word level using the GIZA++ system and the “grow-diag-final-and” heuristics. A 5-gram language model was trained on the Xinhua portion of the English Gigaword corpus in addition to the target-side of the bilingual data. This language model was used in both the baseline and our improved systems. For our skeletal language model, we trained a 5-gram language model on the target-side of the

bilingual data by generalizing non-skeleton segments to Xs. We used the newswire portion of the NIST MT06 evaluation data as our development set, and used the evaluation data of MT04 and MT05 as our test sets. We chose the default feature set of the NiuTrans.Phrase engine for building the baseline, including phrase translation probabilities, lexical weights, a 5-gram language model, word and phrase bonuses, a ME-based lexicalized reordering model. All feature weights were learned using minimum error rate training (Och, 2003).

Our skeleton identification system was built using the t3 toolkit² which implements a state-of-the-art sentence simplification system. We used the NEU Chinese sentence simplification (NEUCSS) corpus as our training data (Zhang et al., 2013). It contains the annotation of sentence skeleton on the Chinese-language side of the Penn Parallel Chinese-English Treebank (LD-C2003E07). We trained our system using the Parts 1-8 of the NEUCSS corpus and obtained a 65.2% relational F1 score and 63.1% compression rate in held-out test (Part 10). For comparison, we also manually annotated the MT development and test data with skeleton information according to the annotation standard provided within NEUCSS.

3.2 Results

Table 1 shows the case-insensitive IBM-version BLEU and TER scores of different systems. We see, first of all, that the MT system benefits from our approach in most cases. In both the manual and automatic identification of sentence skeleton (rows 2 and 4), there is a significant improvement on the “All” data set. However, using different skeleton identification results for training and inference (row 3) does not show big improvements due to the data inconsistency problem.

Another interesting question is whether the skeletal language model really contributes to the improvements. To investigate it, we removed the skeletal language model from our skeleton-based translation system (with automatic skeleton identification on both the development and test sets). Seen from row $-lm^\tau$ of Table 1, the removal of the skeletal language model results in a significant drop in both BLEU and TER performance. It indicates that this language model is very beneficial to our system. For comparison, we removed

²<http://staffwww.dcs.shef.ac.uk/people/T.Cohn/t3/>

system	Entry		MT06 (Dev)		MT04		MT05		All	
	dev-skel	test-skel	BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER
baseline	-	-	35.06	60.54	38.53	61.15	34.32	62.82	36.64	61.54
SBMT	manual	manual	35.71	59.60	38.99	60.67	35.35	61.60	37.30	60.73
SBMT	manual	auto	35.72	59.62	38.75	61.16	35.02	62.20	37.03	61.19
SBMT	auto	auto	35.57	59.66	39.21	60.59	35.29	61.89	37.33	60.80
$-lm^\tau$	auto	auto	35.23	60.17	38.86	60.78	34.82	62.46	36.99	61.16
$-m^\tau$	auto	auto	35.50	59.69	39.00	60.69	35.10	62.03	37.12	60.90
s-space	-	-	35.00	60.50	38.39	61.20	34.33	62.90	36.57	61.58
s-feat.	-	-	35.16	60.50	38.60	61.17	34.25	62.88	36.70	61.58

Table 1: BLEU4[%] and TER[%] scores of different systems. Boldface means a significant improvement ($p < 0.05$). SBMT means our skeleton-based MT system. $-lm^\tau$ (or $-m^\tau$) means that we remove the skeletal language model (or translation model) from our proposed approach. s-space means that we restrict the baseline system to the search space of skeleton-consistent derivations. s-feat. means that we introduce an indicator feature for skeleton-consistent derivations into the baseline system.

the skeleton-based translation model from our system as well. Row $-m^\tau$ of Table 1 shows that the skeleton-based translation model can contribute to the overall improvement but there is no big differences between baseline and $-m^\tau$.

Apart from showing the effects of the skeleton-based model, we also studied the behavior of the MT system under the different settings of search space. Row s-space of Table 1 shows the BLEU and TER results of restricting the baseline system to the space of skeleton-consistent derivations, i.e., we remove both the skeleton-based translation model and language model from the SBMT system. We see that the limited search space is a little harmful to the baseline system. Further, we regarded skeleton-consistent derivations as an indicator feature and introduced it into the baseline system. Seen from row s-feat., this feature does not show promising improvements. These results indicate that the real improvements are due to the skeleton-based model/features used in this work, rather than the "well-formed" derivations.

4 Related Work

Skeleton is a concept that has been used in several sub-areas in MT for years. For example, in confusion network-based system combination it refers to the backbone hypothesis for building confusion networks (Rosti et al., 2007; Rosti et al., 2008); Liu et al. (2011) regard skeleton as a shortened sentence after removing some of the function words for better word deletion. In contrast, we define sentence skeleton as the key segments of a sentence and develop a new MT approach based on this information.

There are some previous studies on the use of sentence skeleton or related information in MT (Mellebeek et al., 2006a; Mellebeek et al., 2006b; Owczarzak et al., 2006). In spite of their good ideas of using skeleton information, they did not model the skeleton-based translation problem in modern SMT pipelines. Our work is a further step towards the use of sentence skeleton in MT. More importantly, we develop a complete approach to this issue and show its effectiveness in a state-of-the-art MT system.

5 Conclusion and Future Work

We have presented a simple but effective approach to integrating the sentence skeleton information into a phrase-based system. The experimental results show that the proposed approach achieves very promising BLEU improvements and TER reductions on the NIST evaluation data. In our future work we plan to investigate methods of integrating both syntactic models (for skeleton translation) and phrasal models (for full translation) in our system. We also plan to study sophisticated reordering models for skeleton translation, rather than reusing the baseline reordering model which is learned on the full sentences.

Acknowledgements

This work was supported in part by the National Science Foundation of China (Grants 61272376 and 61300097), and the China Postdoctoral Science Foundation (Grant 2013M530131). The authors would like to thank the anonymous reviewers for their pertinent and insightful comments.

References

- David Chiang. 2010. Learning to Translate with Source and Target Syntax. In *Proc. of ACL 2010*, pages 1443-1452.
- James Clarke and Mirella Lapata. 2006. Models for Sentence Compression: A Comparison across Domains, Training Requirements and Evaluation Measures. In *Proc. of ACL/COLING 2006*, pages 377-384.
- Trevor Cohn and Mirella Lapata. 2008. Sentence Compression Beyond Word Deletion. In *Proc. of COLING 2008*, pages 137-144.
- Jason Eisner. 2003. Learning Non-Isomorphic Tree Mappings for Machine Translation. In *Proc. of ACL 2003*, pages 205-208.
- Michel Galley and Kathleen McKeown. 2007. Lexicalized Markov Grammars for Sentence Compression. In *Proc. of HLT:NAACL 2007*, pages 180-187.
- Liang Huang, Kevin Knight and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proc. of AMTA 2006*, pages 66-73.
- Kevin Knight and Daniel Marcu. 2000. Statistical-based summarization-step one: sentence compression. In *Proc. of AAAI 2000*, pages 703-710.
- Philipp Koehn, Franz J. Och and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proc. of NAACL 2003*, pages 48-54.
- Yang Liu, Qun Liu and Shouxun Lin. 2006. Tree-to-String Alignment Template for Statistical Machine Translation. In *Proc. of ACL/COLING 2006*, pages 609-616.
- Shujie Liu, Chi-Ho Li and Ming Zhou. 2011. Statistic Machine Translation Boosted with Spurious Word Deletion. In *Proc. of Machine Translation Summit XIII*, pages 72-79.
- Yuval Marton and Philip Resnik. 2008. Soft Syntactic Constraints for Hierarchical Phrased-Based Translation. In *Proc. of ACL:HLT 2008*, pages 1003-1011.
- Bart Mellebeek, Karolina Owczarzak, Josef van Genabith and Andy Way. 2006. Multi-Engine Machine Translation by Recursive Sentence Decomposition. In *Proc. of AMTA 2006*, pages 110-118.
- Bart Mellebeek, Karolina Owczarzak, Declan Groves, Josef Van Genabith and Andy Way. 2006. A Syntactic Skeleton for Statistical Machine Translation. In *Proc. of EAMT 2006*, pages 195-202.
- Franz J. Och, Christoph Tillmann and Hermann Ney. 1999. Improved Alignment Models for Statistical Machine Translation. In *Proc. of EMNLP/VLC 1999*, pages 20-28.
- Franz J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL 2003*, pages 160-167.
- Karolina Owczarzak, Bart Mellebeek, Declan Groves, Josef van Genabith and Andy Way. 2006. Wrapper Syntax for Example-Based Machine Translation. In *Proc. of AMTA2006*, pages 148-155.
- Antti-Veikko I. Rosti, Spyros Matsoukas and Richard Schwartz. 2007. Improved Word-Level System Combination for Machine Translation. In *Proc. of ACL 2007*, pages 312-319.
- Antti-Veikko I. Rosti, Bing Zhang, Spyros Matsoukas, and Richard Schwartz. 2008. Incremental hypothesis alignment for building confusion networks with application to machine translation system combination. In *Proc. of Third Workshop on Statistical Machine Translation*, pages 183 - 186.
- Tong Xiao, Jingbo Zhu, Hao Zhang and Qiang Li 2012. NiuTrans: An Open Source Toolkit for Phrase-based and Syntax-based Machine Translation. In *Proc. of ACL 2012*, system demonstrations, pages 19-24.
- Elif Yamangil and Stuart M. Shieber. 2010. Bayesian Synchronous Tree-Substitution Grammar Induction and Its Application to Sentence Compression. In *Proc. of ACL 2010*, pages 937-947.
- Katsumasa Yoshikawa, Ryu Iida, Tsutomu Hirao and Manabu Okumura. 2012. Sentence Compression with Semantic Role Constraints. In *Proc. of ACL 2012*, pages 349-353.
- Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan and Sheng Li. 2008. A Tree Sequence Alignment-based Tree-to-Tree Translation Model. In *Proc. of ACL:HLT 2008*, pages 559-567.
- Chunliang Zhang, Minghan Hu, Tong Xiao, Xue Jiang, Lixin Shi and Jingbo Zhu. 2013. Chinese Sentence Compression: Corpus and Evaluation. In *Proc. of Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, pages 257-267.

Effective Selection of Translation Model Training Data

Le Liu Yu Hong* Hao Liu Xing Wang Jianmin Yao

School of Computer Science & Technology, Soochow University, China
{20124227052, hongy, 20134227035, 20114227047, jyao}@suda.edu.cn

Abstract

Data selection has been demonstrated to be an effective approach to addressing the lack of high-quality bitext for statistical machine translation in the domain of interest. Most current data selection methods solely use language models trained on a small scale in-domain data to select domain-relevant sentence pairs from general-domain parallel corpus. By contrast, we argue that the relevance between a sentence pair and target domain can be better evaluated by the combination of language model and translation model. In this paper, we study and experiment with novel methods that apply translation models into domain-relevant data selection. The results show that our methods outperform previous methods. When the selected sentence pairs are evaluated on an end-to-end MT task, our methods can increase the translation performance by 3 BLEU points.

1 Introduction

Statistical machine translation depends heavily on large scale parallel corpora. The corpora are necessary priori knowledge for training effective translation model. However, domain-specific machine translation has few parallel corpora for translation model training in the domain of interest. For this, an effective approach is to automatically select and expand domain-specific sentence pairs from large scale general-domain parallel corpus. The approach is named Data Selection. Current data selection methods mostly use language models trained on small scale in-domain data to measure domain relevance and select domain-relevant parallel sentence pairs to expand training corpora. Related work in literature has proven that the expanded corpora can substantially improve the performance of ma-

chine translation (Duh et al., 2010; Haddow and Koehn, 2012).

However, the methods are still far from satisfactory for real application for the following reasons:

- There isn't ready-made domain-specific parallel bitext. So it's necessary for data selection to have significant capability in mining parallel bitext in those assorted free texts. But the existing methods seldom ensure parallelism in the target domain while selecting domain-relevant bitext.
- Available domain-relevant bitext needs keep high domain-relevance at both the sides of source and target language. But it's difficult for current method to maintain two-sided domain-relevance when we aim at enhancing parallelism of bitext.

In a word, current data selection methods can't well maintain both parallelism and domain-relevance of bitext. To overcome the problem, we first propose the method combining translation model with language model in data selection. The language model measures the domain-specific generation probability of sentences, being used to select domain-relevant sentences at both sides of source and target language. Meanwhile, the translation model measures the translation probability of sentence pair, being used to verify the parallelism of the selected domain-relevant bitext.

2 Related Work

The existing data selection methods are mostly based on language model. Yasuda et al. (2008) and Foster et al. (2010) ranked the sentence pairs in the general-domain corpus according to the perplexity scores of sentences, which are computed with respect to in-domain language models. Axelrod et al. (2011) improved the perplexity-based approach and proposed bilingual cross-entropy difference as a ranking function with in- and general-domain language models. Duh et al. (2013) employed the method of (Axelrod et al.,

* Corresponding author

2011) and further explored neural language model for data selection rather than the conventional n-gram language model. Although previous works in data selection (Duh et al., 2013; Koehn and Haddow, 2012; Axelrod et al., 2011; Foster et al., 2010; Yasuda et al., 2008) have gained good performance, the methods which only adopt language models to score the sentence pairs are sub-optimal. The reason is that a sentence pair contains a source language sentence and a target language sentence, while the existing methods are incapable of evaluating the mutual translation probability of sentence pair in the target domain. Thus, we propose novel methods which are based on translation model and language model for data selection.

3 Training Data Selection Methods

We present three data selection methods for ranking and selecting domain-relevant sentence pairs from general-domain corpus, with an eye towards improving domain-specific translation model performance. These methods are based on language model and translation model, which are trained on small in-domain parallel data.

3.1 Data Selection with Translation Model

Translation model is a key component in statistical machine translation. It is commonly used to translate the source language sentence into the target language sentence. However, in this paper, we adopt the translation model to evaluate the translation probability of sentence pair and develop a simple but effective variant of translation model to rank the sentence pairs in the general-domain corpus. The formulations are detailed as below:

$$P(e|f) = \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i) \quad (1)$$

$$R = \sqrt[l_e]{P(e|f)} \quad (2)$$

Where $P(e|f)$ is the translation model, which is IBM Model 1 in this paper, it represents the translation probability of target language sentence e conditioned on source language sentence f . l_e and l_f are the number of words in sentence e and f respectively. $t(e_j|f_i)$ is the translation probability of word e_j conditioned on word f_i and is estimated from the small in-domain parallel data. The parameter ϵ is a constant and is assigned with the value of 1.0. R is the length-normalized IBM Model 1, which is used to score

general-domain sentence pairs. The sentence pair with higher score is more likely to be generated by in-domain translation model, thus, it is more relevant to the in-domain corpus and will be remained to expand the training data.

3.2 Data Selection by Combining Translation and Language model

As described in section 1, the existing data selection methods which only adopt language model to score sentence pairs are unable to measure the mutual translation probability of sentence pairs. To solve the problem, we develop the second data selection method, which is based on the combination of translation model and language model. Our method and ranking function are formulated as follows:

$$P(e, f) = P(e|f) \times P(f) \quad (3)$$

$$R = \sqrt[l_e]{P(e|f)} \times \sqrt[l_f]{P(f)} \quad (4)$$

Where $P(e, f)$ is a joint probability of sentence e and f according to the translation model $P(e|f)$ and language model $P(f)$, whose parameters are estimated from the small in-domain text. R is the improved ranking function and used to score the sentence pairs with the length-normalized translation model $P(e|f)$ and language model $P(f)$. The sentence pair with higher score is more similar to in-domain corpus, and will be picked out.

3.3 Data Selection by Bidirectionally Combining Translation and Language Models

As presented in subsection 3.2, the method combines translation model and language model to rank the sentence pairs in the general-domain corpus. However, it does not evaluate the inverse translation probability of sentence pair and the probability of target language sentence. Thus, we take bidirectional scores into account and simply sum the scores in both directions.

$$R = \sqrt[l_e]{P(e|f)} \times \sqrt[l_f]{P(f)} + \sqrt[l_f]{P(f|e)} \times \sqrt[l_e]{P(e)} \quad (5)$$

Again, the sentence pairs with higher scores are presumed to be better and will be selected to incorporate into the domain-specific training data. This approach makes full use of two translation models and two language models for sentence pairs ranking.

4 Experiments

4.1 Corpora

We conduct our experiments on the Spoken Language Translation English-to-Chinese task. Two corpora are needed for the data selection. The in-domain data is collected from CWMT09, which consists of spoken dialogues in a travel setting, containing approximately 50,000 parallel sentence pairs in English and Chinese. Our general-domain corpus mined from the Internet contains 16 million sentence pairs. Both the in- and general-domain corpora are identically tokenized (in English) and segmented (in Chinese)¹. The details of corpora are listed in Table 1. Additionally, we evaluate our work on the 2004 test set of “863” Spoken Language Translation task (“863” SLT), which consists of 400 English sentences with 4 Chinese reference translations for each. Meanwhile, the 2005 test set of “863” SLT task, which contains 456 English sentences with 4 references each, is used as the development set to tune our systems.

Bilingual Corpus	#sentence		#token	
	Eng	Chn	Eng	Chn
In-domain	50K	50K	360K	310K
General-domain	16M	16M	3933M	3602M

Table 1. Data statistics

4.2 System settings

We use the NiuTrans² toolkit which adopts GIZA++ (Och and Ney, 2003) and MERT (Och, 2003) to train and tune the machine translation system. As NiuTrans integrates the mainstream translation engine, we select hierarchical phrase-based engine (Chiang, 2007) to extract the translation rules and carry out our experiments. Moreover, in the decoding process, we use the NiuTrans decoder to produce the best outputs, and score them with the widely used NIST mt-eval131a³ tool. This tool scores the outputs in several criterions, while the case-insensitive BLEU-4 (Papineni et al., 2002) is used as the evaluation for the machine translation system.

4.3 Translation and Language models

Our work relies on the use of in-domain language models and translation models to rank the sentence pairs from the general-domain bilingual training set. Here, we employ ngram language

model and IBM Model 1 for data selection. Thus, we use the SRI Language Modeling Toolkit (Stolcke, 2002) to train the in-domain 4-gram language model with interpolated modified Kneser-Ney discounting (Chen and Goodman, 1998). The language model is only used to score the general-domain sentences. Meanwhile, we use the language model training scripts integrated in the NiuTrans toolkit to train another 4-gram language model, which is used in MT tuning and decoding. Additionally, we adopt GIZA++ to get the word alignment of in-domain parallel data and form the word translation probability table. This table will be used to compute the translation probability of general-domain sentence pairs.

4.4 Baseline Systems

As described above, by using the NiuTrans toolkit, we have built two baseline systems to fulfill “863” SLT task in our experiments. The In-domain baseline trained on spoken language corpus has 1.05 million rules in its hierarchical-phrase table. While, the General-domain baseline trained on 16 million sentence pairs has a hierarchical phrase table containing 1.7 billion translation rules. These two baseline systems are equipped with the same language model which is trained on large-scale monolingual target language corpus. The BLEU scores of the In-domain and General-domain baseline system are listed in Table 2.

Corpus	Hierarchical phrase	Dev	Test
In-domain	1.05M	15.01	21.99
General-domain	1747M	27.72	34.62

Table 2. Translation performances of In-domain and General-domain baseline systems

The results show that General-domain system trained on a larger amount of bilingual resources outperforms the system trained on the in-domain corpus by over 12 BLEU points. The reason is that large scale parallel corpus maintains more bilingual knowledge and language phenomenon, while small in-domain corpus encounters data sparse problem, which degrades the translation performance. However, the performance of General-domain baseline can be improved further. We use our three methods to refine the general-domain corpus and improve the translation performance in the domain of interest. Thus, we build several contrasting systems trained on refined training data selected by the following different methods.

¹<http://www.nlplab.com/NiuPlan/NiuTrans.YourData.ch.html>

²<http://www.nlplab.com/NiuPlan/NiuTrans.ch.html#download>

³ <http://www.itl.nist.gov/iad/mig/tools>

- **Ngram**: Data selection by 4-gram LMs with Kneser-Ney smoothing. (Axelrod et al., 2011)
- **Neural net**: Data selection by Recurrent Neural LM, with the RNNLM Toolkit. (Duh et al., 2013)
- **Translation Model (TM)**: Data selection with translation model: IBM Model 1.
- **Translation model and Language Model (TM+LM)**: Data selection by combining 4-gram LMs with Kneser-Ney smoothing and IBM model 1(equal weight).
- **Bidirectional TM+LM**: Data selection by bidirectionally combining translation and language models (equal weight).

4.5 Results of Training Data Selection

We adopt five methods for extracting domain-relevant parallel data from general-domain corpus. Using the scoring methods, we rank the sentence pairs of the general-domain corpus and select only the top $N = \{50k, 100k, 200k, 400k, 600k, 800k, 1000k\}$ sentence pairs as refined training data. New MT systems are then trained on these small refined training data. Figure 1 shows the performances of systems trained on selected corpora from the general-domain corpus. The horizontal coordinate represents the number of selected sentence pairs and vertical coordinate is the BLEU scores of MT systems.

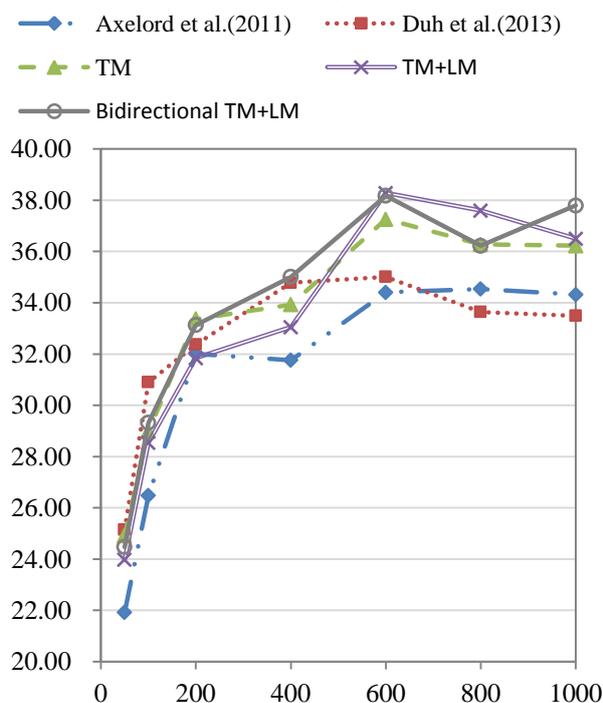


Figure 1. Results of the systems trained on only a subset of the general-domain parallel corpus.

From Figure 1, we conclude that these five data selection methods are effective for domain-specific translation. When top 600k sentence pairs are picked out from general-domain corpus to train machine translation systems, the systems perform higher than the General-domain baseline trained on 16 million parallel data. The results indicate that more training data for translation model is not always better. When the domain-specific bilingual resources are deficient, the domain-relevant sentence pairs will play an important role in improving the translation performance.

Additionally, it turns out that our methods (**TM**, **TM+LM** and **Bidirectional TM+LM**) are indeed more effective in selecting domain-relevant sentence pairs. In the end-to-end SMT evaluation, **TM** selects top 600k sentence pairs of general-domain corpus, but increases the translation performance by 2.7 BLEU points. Meanwhile, the **TM+LM** and **Bidirectional TM+LM** have gained 3.66 and 3.56 BLEU point improvements compared against the general-domain baseline system. Compared with the mainstream methods (**Ngram** and **Neural net**), our methods increase translation performance by nearly 3 BLEU points, when the top 600k sentence pairs are picked out. Although, in the figure 1, our three methods are not performing better than the existing methods in all cases, their overall performances are relatively higher. We therefore believe that combining in-domain translation model and language model to score the sentence pairs is well-suited for domain-relevant sentence pair selection. Furthermore, we observe that the overall performance of our methods is gradually improved. This is because our methods are combining more statistical characteristics of in-domain data in ranking and selecting sentence pairs. The results have proven the effectiveness of our methods again.

5 Conclusion

We present three novel methods for translation model training data selection, which are based on the translation model and language model. Compared with the methods which only employ language model for data selection, we observe that our methods are able to select high-quality domain-relevant sentence pairs and improve the translation performance by nearly 3 BLEU points. In addition, our methods make full use of the limited in-domain data and are easily implemented. In the future, we are interested in applying

our methods into domain adaptation task of statistical machine translation in model level.

Acknowledgments

This research work has been sponsored by two NSFC grants, No.61373097 and No.61272259, and one National Science Foundation of Suzhou (Grants No. SH201212).

Reference

- Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK, July. Association for Computational Linguistics.
- Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 1993, 19(2): 263-311.
- Stanley Chen and Joshua Goodman. 1998. An Empirical Study of Smoothing Techniques for Language Modeling. *Technical Report 10-98, Computer Science Group, Harvard University*.
- Moore Robert C, Lewis William. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*. Association for Computational Linguistics, 2010: 220-224.
- Chiang David. A hierarchical phrase-based model for statistical machine translation. 2005. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages: 263-270. Association for Computational Linguistics.
- Kevin Duh, Graham Neubig, Katsuhito Sudoh and Hajime Tsukada. Adaptation Data Selection using Neural Language Models: Experiments in Machine Translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 678-683, Sofia, Bulgaria, August 4-9 2013.
- Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Analysis of translation model adaptation for statistical machine translation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT) - Technical Papers Track*.
- George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative Instance Weighting for Domain Adaptation in Statistical Machine Translation. *Empirical Methods in Natural Language Processing*.
- Barry Haddow and Philipp Koehn. 2012. Analysing the effect of out-of-domain data on smt systems. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 422–432, Montreal, Canada, June. Association for Computational Linguistics.
- Och, Franz Josef, and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational linguistics* 29.1 (2003): 19-51.
- Och, Franz Josef. Minimum error rate training in statistical machine translation. *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, 2003.
- Philipp Koehn and Barry Haddow. 2012. Towards effective use of training data in statistical machine translation. In *WMT*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *ACL*.
- Andreas Stolcke. 2002. SRILM - An extensible language modeling toolkit. *Spoken Language Processing*.
- Tong Xiao, Jingbo Zhu, Hao Zhang and Qiang Li. NiuTrans: an open source toolkit for phrase-based and syntax-based machine translation. In *Proceedings of the ACL 2012 System Demonstrations*. Association for Computational Linguistics, 2012: 19-24.
- Keiji Yasuda, Ruiqiang Zhang, Hirofumi Yamamoto, and Eiichiro Sumita. 2008. Method of selecting training data to build a compact and efficient translation model. *International Joint Conference on Natural Language Processing*.

Refinements to Interactive Translation Prediction Based on Search Graphs

Philipp Koehn^{◇*}, Chara Tsoukala* and Herve Saint-Amand*

[◇]Center for Speech and Language Processing, The Johns Hopkins University

*School of Informatics, University of Edinburgh

phi@jhu.edu, ctsoukal@inf.ed.ac.uk, hsamand@inf.ed.ac.uk

Abstract

We propose a number of refinements to the canonical approach to interactive translation prediction. By more permissive matching criteria, placing emphasis on matching the last word of the user prefix, and dealing with predictions to partially typed words, we observe gains in both word prediction accuracy (+5.4%) and letter prediction accuracy (+9.3%).

1 Introduction

As machine translation enters the workflow of professional translators, the exact nature of this human-computer interaction is currently an open challenge. Instead of tasking translators to post-edit the output of machine translation systems, a more interactive approach may be more fruitful.

One such idea is interactive translation prediction (Langlais et al., 2000b): While the user writes the translation for a sentence, the system makes suggestions for sequent words. If the user diverges from the suggestions, the system recalculates its prediction, and offers new suggestions. This input modality is familiar to anybody who has used auto-complete functions in text editors, cell phones, or web applications.

The technical challenge is to come up with a method that predicts words that the user will accept. The standard approach to this problem uses the search graph of the machine translation system. Such search graphs may be recomputed in a constraint decoding process restricted to the partial user input (called the *prefix*), but this is often too slow with big models and limited computing resources, so we use static word graphs.

The user prefix is matched against the search graph. If the user prefix cannot be found in the search graph, approximate string matching is used by finding a path with minimal string edit distance, i.e., a path in the graph with the minimal number of insertions, deletions and substitutions to match the user prefix.

This paper presents a number of refinements to extend this approach, by allowing more permissive matching criterion, placing emphasis on matching the last word of the user prefix, and dealing with predictions to partially typed words. We show improvements in word prediction accuracy from 56.1% to 60.5% and letter prediction accuracy from 75.2% to 84.5% on a publicly available benchmark (English-Spanish news translation).

2 Related Work

The interactive machine translation paradigm was first explored in the TransType and TransType2 projects (Langlais et al., 2000a; Foster et al., 2002; Bender et al., 2005; Barrachina et al., 2009). Given the computational cost and need for quick response time, most current word operates on search graphs (Och et al., 2003). Such search graphs can be efficiently represented and processed with finite state tools (Civera et al., 2004). More recently, the approach has been extended to SCFG-based translation models (González-Rubio et al., 2013).

There are several ways the sentence completion predictions can be presented to the user: showing the complete sentence prediction, only a few words, or multiple choices. User actions may be also extended to mouse actions to pinpoint the divergence from an acceptable translation (Sanchis-Trilles et al., 2008), or hand-writing (Alabau et al., 2011) and speech modalities (Cubel et al., 2009).

3 Properties of Core Algorithm

Our implementation of the core algorithm follows closely Koehn (2009). It is a dynamic programming solution that computes the minimal cost to reach each node in the search graph by matching parts of the user prefix. Cost is measured primarily in terms of string edit distance (number of deletions, insertions and substitutions), and secondary in terms of translation model score for the matched path in the graph. Search is done iteratively, with an increasing number of allowable edits.

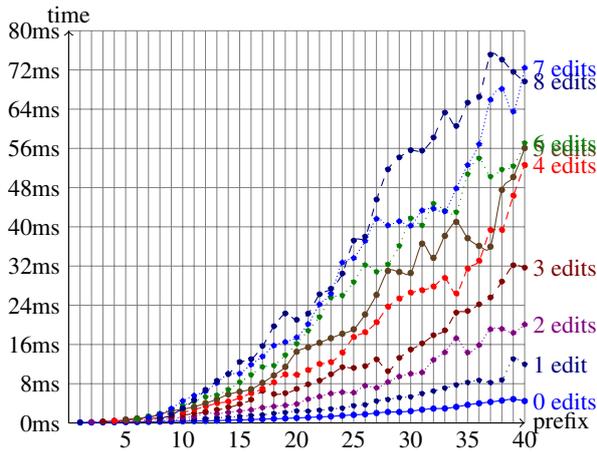


Figure 1: Average response time of baseline method based on length of the prefix and number of edits: The main bottleneck is the string edit distance between prefix and path.

3.1 Experimental Setup

Given the large number of proposed variations of the algorithm, we do not carry out user studies, but rather use a simulated setting. We predict translations that were crafted by manual post-editing of machine translation output. We also use the search graphs of the system that produced the original machine translation output.

Such data has been made available by the CASMACAT project¹. In the project's first field trial², professional translators corrected machine translations of news stories from a competitive English-Spanish machine translation system (Koehn and Haddow, 2012). This test set consists of 24,444 word predictions and 141,662 letter predictions.

3.2 Prediction Speed

Since the interactive translation prediction process is used in an interactive setting where each key stroke of the user may trigger a new request, very fast response time is needed. According to standards in usability engineering

0.1 second is about the limit for having the user feel that the system is reacting instantaneously (Nielsen, 1993).

So, this is the time limit we have to set ourselves to predict the next words of a translator.

What are the main factors that influence processing time in our core algorithm? See Figure 1 for an illustration. We plot processing time against

¹<http://www.casmacat.eu/>

²<http://www.casmacat.eu/uploads/Deliverables/d6.1.pdf>

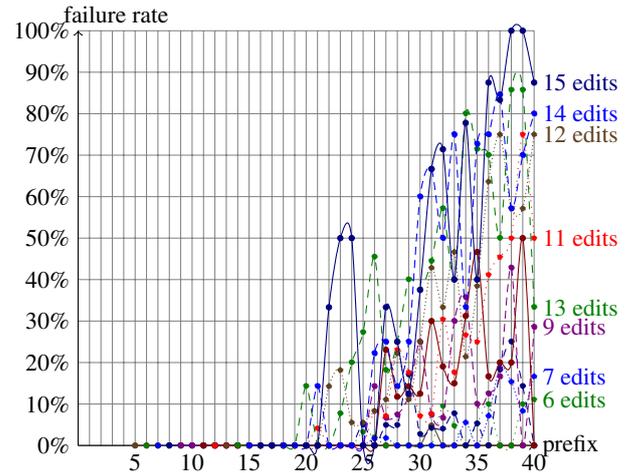


Figure 2: Ratio of prefix matching processes abandoned due to exceeding the 100ms time limit (showing only curves with a minimum of 5 edits).

the length of the user prefix and the string edit distance between the user prefix and the search graph. The graph clearly shows that the main slowdown in processing time occurs when the edit rate increases.

To guarantee a response in 100ms, the algorithm aborts when this time is exceeded and relies on a prediction based on string edit distance against the best path in the graph. The larger the number of edits, the more often this occurs, as Figure 2 shows.

3.3 Accuracy

We are mainly interested in the accuracy of the method: How often does it predict a word that the user accepts? There is a trade-off between speed and accuracy.

One way we can balance this trade-off is by removing nodes from the search graph. By threshold pruning (Sanchis-Trilles and Ortiz-Martínez, 2014), we remove nodes from the search graph that are only part of paths that are worse than the best path by a specified score difference.

See Table 1 how the choice of the score difference threshold impacts failure rate and accuracy. A wider threshold has the potential to achieve better results (if we allow for up to 1 second of processing time), but with the constraint of 100ms response time, the optimum is with a threshold of 0.4. Wider thresholds lead to a higher failure rate, causing overall lower accuracy.

Threshold	100ms Max		1000ms Max	
	Acc.	Fail	Acc.	Fail
0.3	55.8%	4.5%	56.9%	0.0%
0.4	56.1%	6.5%	58.0%	0.0%
0.5	55.9%	9.0%	58.8%	0.0%
0.6	55.5%	11.6%	59.4%	0.0%
0.8	54.4%	17.1%	59.4%	0.0%
1.0	52.7%	21.7%	58.6%	6.5%

Table 1: Impact of threshold pruning on search accuracy and failure rate (i.e., failure to complete search in given time and resorting to matching against best translation).

4 Refinements

We now introduce a number of refinements over the core method. Given the constraints established in the previous section (maximum response time of 100ms, pruning threshold 0.4), we set out to improve accuracy.

4.1 Matching Last Word

The first idea is that it is more important to match the last word of the user prefix than having mismatches in earlier words. We attempt to find the last word in the predicted path either before or after the optimal matching position according to string edit distance.

We combine the matched path in the prefix with the optimal suffix, and search for the last user prefix word within a window. This means that we either move words from the suffix to the prefix or the other way around, without changing the overall string along the path.

Table 2 shows the impact on accuracy for different window sizes. While we expected some gains by checking for the word somewhere around the optimal position in the predicted path, we do see significant gains by not placing any restrictions to where the word can be found, except for a bias to less distant positions. For instance, examining a window of up to 3 words gives us a word prediction accuracy of 57.2% versus the 56.1% baseline. Finding the last word anywhere boosts performance to 59.1%.

The table also reports accuracy numbers when we allow the process to run up to 1 second — which is basically an exhaustive search but not practically useful. These numbers shed some light on why an unlimited window size in matching the last word helps: the gains come partially from the cases where the initial search fails. Finding the last user word anywhere in the machine transla-

Window	100ms Max	1000ms Max
baseline	56.1%	58.0%
1 word	56.6%	58.4%
2 words	56.9%	58.6%
3 words	57.2%	58.9%
5 words	57.8%	59.3%
anywhere	59.1%	59.5%

Table 2: Search for the last prefix word in a window around the predicted position in the matched path.

Word Matching	100ms Max	1000ms Max
baseline	59.1%	59.5%
case-insensitive	58.7%	59.4%

Table 3: Search with case-insensitive word matching (say, *University* and *university*).

tion output is a better fallback than computing optimal string edit distance. Analysis of the data suggests that gains mainly come from large length mismatches between user translation and machine translation, even in the case of first pass searches.

4.2 Case-Insensitive Matching

Some mismatches between words matter less than others. For instance, if the user prefix differs only in casing from the machine translation (say, *University* instead of *university*), then we may still want to treat that as a word match in our algorithm. However, as Table 3 shows, allowing case-insensitive matching leads to lower accuracy (58.7% vs. 59.1%).

A major reason is computational cost. The most inner loop in the algorithm compares words. This is optimized by representing words as integers. However, if we allow case-insensitive matching, this simple method does not work anymore. We do precompute approximate word matches and store matching words identifiers in a hash map, but still the ratio of searches that do not complete in 100ms increases from 6.5% to 9.7%. By extending the allowable time to 1 second, the accuracy gap is reduced to 0.1%.

4.3 Approximate Word Matching

When a word in the user translation differs from a word in the decoder search graph only by a few letters, then it should be considered a lesser error than substitutions of completely different words. Such word differences may be due to casing, morphological variants, or spelling inconsistencies.

We compute word dissimilarity by computing

Max. Dissimilarity	100ms Max.	1000ms Max.
baseline	59.1%	59.5%
30%	60.2%	61.0%
20%	60.4%	61.3%
10%	60.6%	61.5%

Table 4: Counting substitutions between similar words as half an error. Dissimilarity is measured as letter edit distance

Min Stem / Max Suffix	100ms	1000ms
baseline	59.1%	59.5%
4 / 3	59.4%	60.1%
3 / 3	59.5%	60.2%
2 / 3	59.5%	60.3%

Table 5: Counting substitutions between morphological variants as half an error. Morphological variance is approximated by requiring a minimum number of initial letters to match and a maximum of final letters to differ.

the ratio of letter edit operations to the length of the shorter word.³ We now set a threshold for maximum dissimilarity, under which mismatched words are considered only half the edit cost of other edit operations.

Table 4 shows that we get significantly higher word prediction accuracy than with the baseline approach (up to 60.6% vs. 59.1%), and the best performance with a 10% threshold. We observe the same computational problem as in the previous section (about 9.2% first pass failures, vs. 6.5%), reflected in a higher accuracy gap for 100ms and 1000ms time limits.

4.4 Stemmed Matching

We suspected that the main benefit of approximate word matching is the better handling of morphological variants. In Spanish, this mainly constitutes itself as different word endings. Thus, we redefine our word dissimilarity measure by consider words similar, if they agree in at least a number of leading letters (presumably the stem), and may differ in at most a number of trailing letters (presumably the morpheme).

Table 5 shows that this is successful in increasing the word prediction rate (59.5% vs. 59.1%) but not as much as with the more general approximate word matching in the previous section (recall: 60.6%).

³For instance, if a 6 letter word and a 4 letter word can be matched with two deletions and one substitution, then the dissimilarity score is $\frac{3}{4} = .75$.

#	Method	Word Acc.	Letter Acc.
1	baseline	56.0%	75.2%
2	1+matching last word	59.0%	80.6%
3	2+case insensitive	58.7%	80.4%
4	2+dissimilarity 10%	60.5%	80.6%
5	2+stem 2/3	59.4%	80.5%
6	4+desperate	60.5%	84.5%

Table 6: Extending the approach to word completion. Impact of refinements of letter prediction accuracy with additional desperate word matching against the entire vocabulary.

5 Word Completion

Besides word prediction, word completion is also a useful feature in an interactive translation tool. When the machine translation system decides for *college* over *university*, but the user types the letter *u*, it should change its prediction.

To enable word completion in the canonical algorithm, we allow matching of the final user word (if not followed by a space character) as a prefix of any word as a zero cost operation. The predicted suffix that is returned to the user then starts with the remaining letters of the word in the path.

Table 6 shows that the refinements that helped sentence completion also benefit word completion. From a baseline accuracy of 75.2% correctly predicted letters, we reach up to 80.6%. Note that the baseline word prediction accuracy is slightly lower (56.0% vs. 56.1%) than in the previous experiments, since the previously correctly matched last word may be mistaken as the prefix of another word.

We add an additional refinement to this task: If the potentially incomplete final word of the user prefix cannot be found in the predicted path, then we explore the entire vocabulary from the unpruned search graph for completions. If multiple words match, the one with the highest path score is used. This *desperate* word completion method gives significant gains (84.5% over 80.6%).

6 Conclusion and Future Work

We observe most improvements by a focus on the last word of the user prefix and approximate word matching. This suggests that there may be additional gains by a stronger focus on the tail of the user prefix. Also, the findings from the time/productivity tradeoffs indicate that more time efficient algorithms and implementations should be explored.

Acknowledgements

This work was supported under the CASMACAT project (grant agreement N° 287576) by the European Union 7th Framework Programme (FP7/2007-2013).

References

- Alabau, V., Sanchis, A., and Casacuberta, F. (2011). Improving on-line handwritten recognition using translation models in multimodal interactive machine translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 389–394, Portland, Oregon, USA. Association for Computational Linguistics.
- Barrachina, S., Bender, O., Casacuberta, F., Civera, J., Cubel, E., Khadivi, S., Lagarda, A., Ney, H., Tomás, J., Vidal, E., and Vilar, J.-M. (2009). Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1).
- Bender, O., Hasan, S., Vilar, D., Zens, R., and Ney, H. (2005). Comparison of generation strategies for interactive machine translation. In *Proceedings of the 10th Conference of the European Association for Machine Translation (EAMT)*, Budapest.
- Civera, J., Cubel, E., Lagarda, A. L., Picó, D., González, J., Vidal, E., Casacuberta, F., Vilar, J. M., and Barrachina, S. (2004). From machine translation to computer assisted translation using finite-state models. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 349–356, Barcelona, Spain. Association for Computational Linguistics.
- Cubel, E., Khadivi, S., Lagarda, A., Ney, H., Toms, J., Vidal, E., and Vilar, J.-M. (2009). Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1).
- Foster, G., Langlais, P., and Lapalme, G. (2002). User-friendly text prediction for translators. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 148–155, Philadelphia. Association for Computational Linguistics.
- González-Rubio, J., Ortíz-Martínez, D., Benedí, J.-M., and Casacuberta, F. (2013). Interactive machine translation using hierarchical translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 244–254, Seattle, Washington, USA. Association for Computational Linguistics.
- Koehn, P. (2009). A process study of computer-aided translation. *Machine Translation*, 23(4):241–263.
- Koehn, P. and Haddow, B. (2012). Towards effective use of training data in statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 363–367, Montreal, Canada. Association for Computational Linguistics.
- Langlais, P., Foster, G., and Lapalme, G. (2000a). Transtype: a computer-aided translation typing system. In *Proceedings of the ANLP-NAACL 2000 Workshop on Embedded Machine Translation Systems*.
- Langlais, P., Foster, G., and Lapalme, G. (2000b). Unit completion for a computer-aided translation typing system. In *Proceedings of Annual Meeting of the North American Chapter of the Association of Computational Linguistics (NAACL)*.
- Nielsen, J. (1993). *Usability Engineering*. Morgan Kaufmann.
- Och, F. J., Zens, R., and Ney, H. (2003). Efficient search for interactive statistical machine translation. In *Proceedings of Meeting of the European Chapter of the Association of Computational Linguistics (EACL)*.
- Sanchis-Trilles, G. and Ortiz-Martínez, D. (2014). Efficient wordgraph pruning for interactive translation prediction. In *Annual Conference of the European Association for Machine Translation (EAMT)*.
- Sanchis-Trilles, G., Ortiz-Martínez, D., Civera, J., Casacuberta, F., Vidal, E., and Hoang, H. (2008). Improving interactive machine translation via mouse actions. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 485–494, Honolulu, Hawaii. Association for Computational Linguistics.

Cross-lingual Model Transfer Using Feature Representation Projection

Mikhail Kozhevnikov

MMCI, University of Saarland
Saarbrücken, Germany

mkozhevn@mmci.uni-saarland.de

Ivan Titov

ILLC, University of Amsterdam
Amsterdam, Netherlands

titov@uva.nl

Abstract

We propose a novel approach to cross-lingual model transfer based on *feature representation projection*. First, a compact feature representation relevant for the task in question is constructed for either language independently and then the mapping between the two representations is determined using parallel data. The target instance can then be mapped into the source-side feature representation using the derived mapping and handled directly by the source-side model. This approach displays competitive performance on model transfer for semantic role labeling when compared to direct model transfer and annotation projection and suggests interesting directions for further research.

1 Introduction

Cross-lingual model transfer approaches are concerned with creating statistical models for various tasks for languages poor in annotated resources, utilising resources or models available for these tasks in other languages. That includes approaches such as *direct model transfer* (Zeman and Resnik, 2008) and *annotation projection* (Yarowsky et al., 2001). Such methods have been successfully applied to a variety of tasks, including POS tagging (Xi and Hwa, 2005; Das and Petrov, 2011; Täckström et al., 2013), syntactic parsing (Ganchev et al., 2009; Smith and Eisner, 2009; Hwa et al., 2005; Durrett et al., 2012; Søgaard, 2011), semantic role labeling (Padó and Lapata, 2009; Annesi and Basili, 2010; Tonelli and Pianta, 2008; Kozhevnikov and Titov, 2013) and others.

Direct model transfer attempts to find a shared feature representation for samples from the two languages, usually generalizing and abstracting away from language-specific representations.

Once this is achieved, instances from both languages can be mapped into this space and a model trained on the source-language data directly applied to the target language. If parallel data is available, it can be further used to enforce model agreement on this data to adjust for discrepancies between the two languages, for example by means of *projected transfer* (McDonald et al., 2011).

The shared feature representation depends on the task in question, but most often each aspect of the original feature representation is handled separately. Word types, for example, may be replaced by cross-lingual word clusters (Täckström et al., 2012) or cross-lingual distributed word representations (Klementiev et al., 2012). Part-of-speech tags, which are often language-specific, can be converted into universal part-of-speech tags (Petrov et al., 2012) and morpho-syntactic information can also be represented in a unified way (Zeman et al., 2012; McDonald et al., 2013; Tsarfaty, 2013). Unfortunately, the design of such representations and corresponding conversion procedures is by no means trivial.

Annotation projection, on the other hand, does not require any changes to the feature representation. Instead, it operates on translation pairs, usually on sentence level, applying the available source-side model to the source sentence and transferring the resulting annotations through the word alignment links to the target one. The quality of predictions on source sentences depends heavily on the quality of parallel data and the domain it belongs to (or, rather, the similarity between this domain and that of the corpus the source-language model was trained on). The transfer itself also introduces errors due to translation shifts (Cyrus, 2006) and word alignment errors, which may lead to inaccurate predictions. These issues are generally handled using heuristics (Padó and Lapata, 2006) and filtering, for example based on alignment coverage (van der Plas et al., 2011).

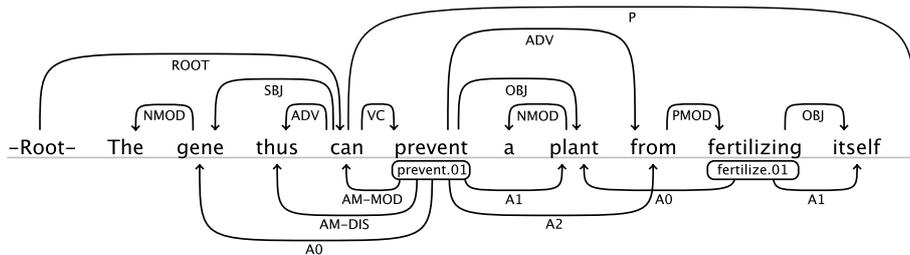


Figure 1: Dependency-based semantic role labeling example. The top arcs depict dependency relations, the bottom ones – semantic role structure. Rendered with <https://code.google.com/p/whatswrong/>.

1.1 Motivation

The approach proposed here, which we will refer to as *feature representation projection* (FRP), constitutes an alternative to direct model transfer and annotation projection and can be seen as a compromise between the two.

It is similar to direct transfer in that we also use a shared feature representation. Instead of designing this representation manually, however, we create compact monolingual feature representations for source and target languages separately and automatically estimate the mapping between the two from parallel data. This allows us to make use of language-specific annotations and account for the interplay between different types of information. For example, a certain preposition attached to a token in the source language might map into a morphological tag in the target language, which would be hard to handle for traditional direct model transfer other than using some kind of refinement procedure involving parallel data. Note also that any such refinement procedure applicable to direct transfer would likely work for FRP as well.

Compared to annotation projection, our approach may be expected to be less sensitive to parallel data quality, since we do not have to commit to a particular prediction on a given instance from parallel data. We also believe that FRP may profit from using other sources of information about the correspondence between source and target feature representations, such as dictionary entries, and thus have an edge over annotation projection in those cases where the amount of parallel data available is limited.

2 Evaluation

We evaluate feature representation projection on the task of dependency-based semantic role labeling (SRL) (Hajič et al., 2009).

This task consists in identifying predicates and their arguments in sentences and assigning each argument a semantic role with respect to its predicate (see figure 1). Note that only a single word – the syntactic head of the argument phrase – is marked as an argument in this case, as opposed to constituent- or span-based SRL (Carreras and Màrquez, 2005). We focus on the assignment of semantic roles to identified arguments.

For the sake of simplicity we cast it as a multi-class classification problem, ignoring the interaction between different arguments in a predicate. It is well known that such interaction plays an important part in SRL (Punyakanok et al., 2008), but it is not well understood which kinds of interactions are preserved across languages and which are not. Also, should one like to apply constraints on the set of semantic roles in a given predicate, or, for example, use a reranker (Björkelund et al., 2009), this can be done using a factorized model obtained by cross-lingual transfer.

In our setting, each *instance* includes the word type and part-of-speech and morphological tags (if any) of argument token, its parent and corresponding predicate token, as well as their dependency relations to their respective parents. This representation is further denoted ω_0 .

2.1 Approach

We consider a pair of languages (L^s, L^t) and assume that an annotated training set $D_T^s = \{(x^s, y^s)\}$ is available in the source language as well as a parallel corpus of instance pairs $D^{st} = \{(x^s, x^t)\}$ and a target dataset $D_E^t = \{x^t\}$ that needs to be labeled.

We design a pair of intermediate compact monolingual feature representations ω_1^s and ω_1^t and models M_s and M_t to map source and target samples x^s and x^t from their original representations, ω_0^s and ω_0^t , to the new ones. We use the par-

allel instances in the new feature representation

$$\bar{D}^{st} = \left\{ \left(x_1^s, x_1^t \right) \right\} = \left\{ \left(M_s(x^s), M_t(x^t) \right) \right\}$$

to determine the mapping M_{ts} (usually, linear) between the two spaces:

$$M_{ts} = \underset{(x_1^s, x_1^t \in \bar{D}^{st})}{\operatorname{argmax}_M} \sum \left\| x_1^s - M(x_1^t) \right\|_2$$

Then a classification model M_y is trained on the source training data

$$\bar{D}_T^s = \left\{ \left(x_1^s, y^s \right) \right\} = \left\{ \left(M_s(x^s), y^s \right) \right\}$$

and the labels are assigned to the target samples $x^t \in D_E^t$ using a composition of the models:

$$y^t = M_y(M_{ts}(M_t(x^t)))$$

2.2 Feature Representation

Our objective is to make the feature representation sufficiently compact that the mapping between source and target feature spaces could be reliably estimated from a limited amount of parallel data, while preserving, insofar as possible, the information relevant for classification.

Estimating the mapping directly from raw categorical features (ω_0) is both computationally expensive and likely inaccurate – using one-hot encoding the feature vectors in our experiments would have tens of thousands of components. There is a number of ways to make this representation more compact. To start with, we replace word types with corresponding neural language model representations estimated using the skip-gram model (Mikolov et al., 2013a). This corresponds to M_s and M_t above and reduces the dimension of the feature space, making direct estimation of the mapping practical. We will refer to this representation as ω_1 .

To go further, one can, for example, apply dimensionality reduction techniques to obtain a more compact representation of ω_1 by eliminating redundancy or define auxiliary tasks and produce a vector representation useful for those tasks. In source language, one can even directly tune an intermediate representation for the target problem.

2.3 Baselines

As mentioned above we compare the performance of this approach to that of direct transfer and annotation projection. Both baselines are using the

same set of features as the proposed model, as described earlier.

The shared feature representation for direct transfer is derived from ω_0 by replacing language-specific part-of-speech tags with universal ones (Petrov et al., 2012) and adding cross-lingual word clusters (Täckström et al., 2012) to word types. The word types themselves are left as they are in the source language and replaced with their gloss translations in the target one (Zeman and Resnik, 2008). In English-Czech and Czech-English we also use the dependency relation information, since the annotations are partly compatible.

The annotation projection baseline implementation is straightforward. The source-side instances from a parallel corpus are labeled using a classifier trained on source-language training data and transferred to the target side. The resulting annotations are then used to train a target-side classifier for evaluation. Note that predicate and argument identification in both languages is performed using monolingual classifiers and only aligned pairs are used in projection. A more common approach would be to project the whole structure from the source language, but in our case this may give unfair advantage to feature representation projection, which relies on target-side argument identification.

2.4 Tools

We use the same type of log-linear classifiers in the model itself and the two baselines to avoid any discrepancy due to learning procedure. These classifiers are implemented using PYLEARN2 (Goodfellow et al., 2013), based on THEANO (Bergstra et al., 2010). We also use this framework to estimate the linear mapping M_{ts} between source and target feature spaces in FRP.

The 250-dimensional word representations for ω_1 are obtained using WORD2VEC tool. Both monolingual data and that from the parallel corpus are included in the training. In Mikolov et al. (2013b) the authors consider embeddings of up to 800 dimensions, but we would not expect to benefit as much from larger vectors since we are using a much smaller corpus to train them. We did not tune the size of the word representation to our task, as this would not be appropriate in a cross-lingual transfer setup, but we observe that the classifier is relatively robust to their dimension when evalu-

ated on source language – in our experiments the performance of the monolingual classifier does not improve significantly if the dimension is increased past 300 and decreases only by a small margin (less than one absolute point) if it is reduced to 100. It should be noted, however, that the dimension that is optimal in this sense is not necessarily the best choice for FRP, especially if the amount of available parallel data is limited.

2.5 Data

We use two language pairs for evaluation: English-Czech and English-French. In the first case, the data is converted from Prague Czech-English Dependency Treebank 2.0 (Hajič et al., 2012) using the script from Kozhevnikov and Titov (2013). In the second, we use CoNLL 2009 shared task (Hajič et al., 2009) corpus for English and the manually corrected dataset from van der Plas et al. (2011) for French. Since the size of the latter dataset is relatively small – one thousand sentences – we reserve the whole dataset for testing and only evaluate transfer from English to French, but not the other way around. Datasets for other languages are sufficiently large, so we take 30 thousand samples for testing and use the rest as training data. The validation set in each experiment is withheld from the corresponding training corpus and contains 10 thousand samples.

Parallel data for both language pairs is derived from Europarl (Koehn, 2005), which we preprocess using MATE-TOOLS (Björkelund et al., 2009; Bohnet, 2010).

3 Results

The classification error of FRP and the baselines given varying amount of parallel data is reported in figures 2, 3 and 4. The training set for each language is fixed. We denote the two baselines AP (annotation projection) and DT (direct transfer).

The number of parallel instances in these experiments is shown on a logarithmic scale, the values considered are 2, 5, 10, 20 and 50 thousand pairs.

Please note that we report only a single value for direct transfer, since this approach does not explicitly rely on parallel data. Although some of the features – namely, gloss translations and cross-lingual clusters – used in direct transfer are, in fact, derived from parallel data, we consider the effect of this on the performance of direct transfer to be indirect and outside the scope of this work.

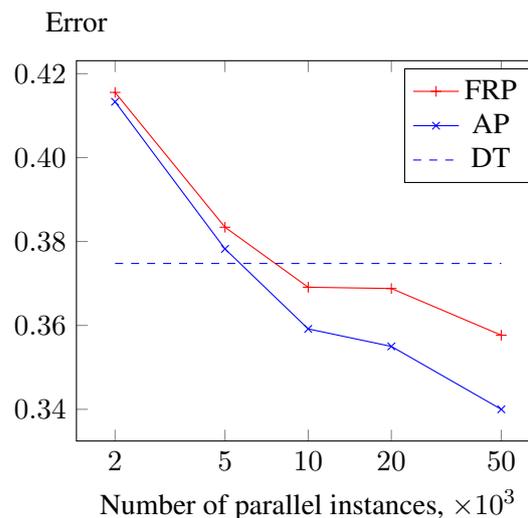


Figure 2: English-Czech transfer results

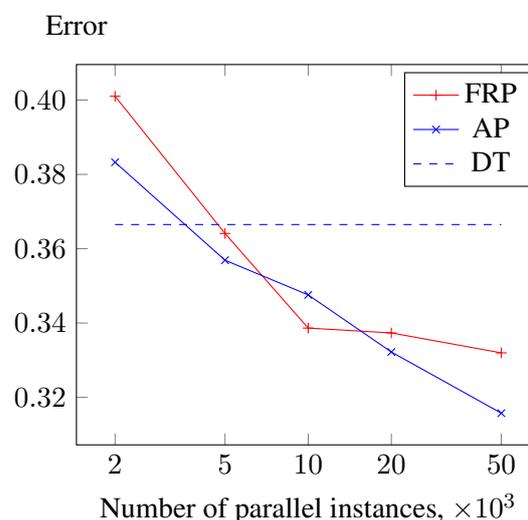


Figure 3: Czech-English transfer results

The rather inferior performance of direct transfer baseline on English-French may be partially attributed to the fact that it cannot rely on dependency relation features, as the corpora we consider make use of different dependency relation inventories. Replacing language-specific dependency annotations with the universal ones (McDonald et al., 2013) may help somewhat, but we would still expect the methods directly relying on parallel data to achieve better results given a sufficiently large parallel corpus.

Overall, we observe that the proposed method with ω_1 representation demonstrates performance competitive to direct transfer and annotation projection baselines.

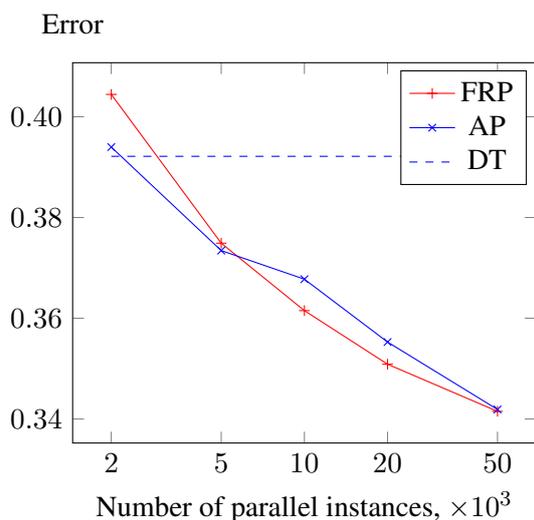


Figure 4: English-French transfer results

4 Additional Related Work

Apart from the work on direct/projected transfer and annotation projection mentioned above, the proposed method can be seen as a more explicit kind of domain adaptation, similar to Titov (2011) or Blitzer et al. (2006).

It is also somewhat similar in spirit to Mikolov et al. (2013b), where a small number of word translation pairs are used to estimate a mapping between distributed representations of words in two different languages and build a word translation model.

5 Conclusions

In this paper we propose a new method of cross-lingual model transfer, report initial evaluation results and highlight directions for its further development.

We observe that the performance of this method is competitive with that of established cross-lingual transfer approaches and its application requires very little manual adjustment – no heuristics or filtering and no explicit shared feature representation design. It also retains compatibility with any refinement procedures similar to projected transfer (McDonald et al., 2011) that may have been designed to work in conjunction with direct model transfer.

6 Future Work

This paper reports work in progress and there is a number of directions we would like to pursue further.

Better Monolingual Representations The representation we used in the initial evaluation does not discriminate between aspects that are relevant for the assignment of semantic roles and those that are not. Since we are using a relatively small set of features to start with, this does not present much of a problem. In general, however, retaining only relevant aspects of intermediate monolingual representations would simplify the estimation of mapping between them and make FRP more robust.

For source language, this is relatively straightforward, as the intermediate representation can be directly tuned for the problem in question using labeled training data. For target language, however, we assume that no labeled data is available and auxiliary tasks have to be used to achieve this.

Alternative Sources of Information The amount of parallel data available for many language pairs is growing steadily. However, cross-lingual transfer methods are often applied in cases where parallel resources are scarce or of poor quality and must be used with care. In such situations an ability to use alternative sources of information may be crucial. Potential sources of such information include dictionary entries or information about the mapping between certain elements of syntactic structure, for example a known part-of-speech tag mapping.

The available parallel data itself may also be used more comprehensively – aligned arguments of aligned predicates, for example, constitute only a small part of it, while the mapping of vector representations of individual tokens is likely to be the same for all aligned words.

Multi-source Transfer One of the strong points of direct model transfer is that it naturally fits the multi-source transfer setting. There are several possible ways of adapting FRP to such a setting. It remains to be seen which one would produce the best results and how multi-source feature representation projection would compare to, for example, multi-source projected transfer (McDonald et al., 2011).

Acknowledgements

The authors would like to acknowledge the support of MMCI Cluster of Excellence and Saarbrücken Graduate School of Computer Science and thank the anonymous reviewers for their suggestions.

References

- Paolo Annesi and Roberto Basili. 2010. Cross-lingual alignment of FrameNet annotations through hidden Markov models. In *Proceedings of the 11th international conference on Computational Linguistics and Intelligent Text Processing, CICLing'10*, pages 12–25. Springer-Verlag.
- James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, Austin, TX.
- Anders Björkelund, Love Hafdell, and Pierre Nugues. 2009. Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 43–48, Boulder, Colorado, June. Association for Computational Linguistics.
- John Blitzer, Ryan McDonal, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proc. Conference on Empirical Methods in Natural Language Processing*, Sydney, Australia.
- Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 89–97, Beijing, China, August.
- Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of CoNLL-2005*, Ann Arbor, MI USA.
- Lea Cyrus. 2006. Building a resource for studying translation shifts. *CoRR*, abs/cs/0606096.
- Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 600–609, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Greg Durrett, Adam Pauls, and Dan Klein. 2012. Syntactic transfer using a bilingual lexicon. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1–11, Jeju Island, Korea, July. Association for Computational Linguistics.
- Kuzman Ganchev, Jennifer Gillenwater, and Ben Taskar. 2009. Dependency grammar induction via bitext projection constraints. 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 369–377, Suntec, Singapore, August. Association for Computational Linguistics.
- Ian J. Goodfellow, David Warde-Farley, Pascal Lamblin, Vincent Dumoulin, Mehdi Mirza, Razvan Pascanu, James Bergstra, Frédéric Bastien, and Yoshua Bengio. 2013. Pylearn2: a machine learning research library. *CoRR*, abs/1308.4214.
- Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado.
- Jan Hajič, Eva Hajičová, Jarmila Panevová, Petr Sgall, Ondřej Bojar, Silvie Cinková, Eva Fučíková, Marie Mikulová, Petr Pajas, Jan Popelka, Jiří Semecký, Jana Šindlerová, Jan Štěpánek, Josef Toman, Zdeňka Urešová, and Zdeněk Žabokrtský. 2012. Announcing Prague Czech-English dependency treebank 2.0. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, May. European Language Resources Association (ELRA).
- Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel text. *Natural Language Engineering*, 11(3):311–325.
- Alexandre Klementiev, Ivan Titov, and Binod Bhat-tarai. 2012. Inducing crosslingual distributed representations of words. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, Bombay, India.
- Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand. AAMT.
- Mikhail Kozhevnikov and Ivan Titov. 2013. Cross-lingual transfer of semantic role labeling models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1190–1200, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11*, pages 62–72, Edinburgh, United Kingdom. Association for Computational Linguistics.

- Ryan McDonald, Joakim Nivre, Yvonne Quirnbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.
- Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013b. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.
- Sebastian Padó and Mirella Lapata. 2006. Optimal constituent alignment with edge covers for semantic projection. In *Proc. 44th Annual Meeting of Association for Computational Linguistics and 21st International Conf. on Computational Linguistics, ACL-COLING 2006*, pages 1161–1168, Sydney, Australia.
- Sebastian Padó and Mirella Lapata. 2009. Cross-lingual annotation projection for semantic roles. *Journal of Artificial Intelligence Research*, 36:307–340.
- Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of LREC*, May.
- 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.
- David A Smith and Jason Eisner. 2009. Parser adaptation and projection with quasi-synchronous grammar features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 822–831. Association for Computational Linguistics.
- Anders Søgaard. 2011. Data point selection for cross-language adaptation of dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, volume 2 of *HLT '11*, pages 682–686, Portland, Oregon. Association for Computational Linguistics.
- Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proc. of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*, pages 477–487, Montréal, Canada.
- Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. *Transactions of the Association for Computational Linguistics*, 1:1–12.
- Ivan Titov. 2011. Domain adaptation by constraining inter-domain variability of latent feature representation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 62–71, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Sara Tonelli and Emanuele Pianta. 2008. Frame information transfer from English to Italian. In *Proceedings of LREC 2008*.
- Reut Tsarfaty. 2013. A unified morpho-syntactic scheme of stanford dependencies. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 578–584, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Lonneke van der Plas, Paola Merlo, and James Henderson. 2011. Scaling up automatic cross-lingual semantic role annotation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, HLT '11*, pages 299–304, Portland, Oregon, USA. Association for Computational Linguistics.
- Chenhai Xi and Rebecca Hwa. 2005. A backoff model for bootstrapping resources for non-english languages. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 851–858, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.
- David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the first international conference on Human language technology research*, pages 1–8. Association for Computational Linguistics.
- Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*, pages 35–42, Hyderabad, India, January. Asian Federation of Natural Language Processing.
- Daniel Zeman, David Mareček, Martin Popel, Loganathan Ramasamy, Jan Štěpánek, Zdeněk Žabokrtský, and Jan Hajič. 2012. Hamletd: To parse or not to parse? In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may. European Language Resources Association (ELRA).

Cross-language and Cross-encyclopedia Article Linking Using Mixed-language Topic Model and Hypernym Translation

Yu-Chun Wang
Department of CSIE
National Taiwan University
Taipei, Taiwan
d97023@csie.ntu.edu.tw

Chun-Kai Wu
Department of CSIE
National Tsinghua University
Hsinchu, Taiwan
s102065512@m102.nthu.edu.tw

Richard Tzong-Han Tsai*
Department of CSIE
National Central University
Chungli, Taiwan
tchtsai@csie.ncu.edu.tw

Abstract

Creating cross-language article links among different online encyclopedias is now an important task in the unification of multilingual knowledge bases. In this paper, we propose a cross-language article linking method using a mixed-language topic model and hypernym translation features based on an SVM model to link English Wikipedia and Chinese Baidu Baike, the most widely used Wiki-like encyclopedia in China. To evaluate our approach, we compile a data set from the top 500 Baidu Baike articles and their corresponding English Wiki articles. The evaluation results show that our approach achieves 80.95% in MRR and 87.46% in recall. Our method does not heavily depend on linguistic characteristics and can be easily extended to generate cross-language article links among different online encyclopedias in other languages.

1 Introduction

Online encyclopedias are among the most frequently used Internet services today. One of the largest and best known online encyclopedias is Wikipedia. Wikipedia has many language versions, and articles in one language contain hyperlinks to corresponding pages in other languages. However, the coverage of different language versions of Wikipedia is very inconsistent. Table 1 shows the statistics of inter-language link pages in the English and Chinese editions in February 2014. The total number of Chinese articles is about one-quarter of English ones, and only 2.3% of English articles have inter-language links to their Chinese versions.

*corresponding author

	Articles	Inter-language Links	Ratio	
zh	755,628	zh2en	486,086	64.3%
en	4,470,246	en2zh	106,729	2.3%

Table 1: Inter-Language Links in Wikipedia

However, there are alternatives to Wikipedia for some languages. In China, for example Baidu Baike and Hudong are the largest encyclopedia sites, containing more than 6.2 and 7 million Chinese articles respectively. Similarly, in Korea, Naver Knowledge Encyclopedia has a large presence.

Since alternative encyclopedias like Baidu Baike are larger (by article count) and growing faster than the Chinese Wikipedia, it is worthwhile to investigate creating cross-language links among different online encyclopedias. Several works have focused on creating cross-language links between Wikipedia language versions (Oh et al., 2008; Sorg and Cimiano, 2008) or finding a cross-language link for each entity mention in a Wikipedia article, namely Cross-Language Link Discovery (CLLD) (Tang et al., 2013; McNamee et al., 2011). These works were able to exploit the link structure and metadata common to all Wikipedia language versions. However, when linking between different online encyclopedia platforms this is more difficult as many of these structural features are different or not shared. To date, little research has been done into linking between encyclopedias on different platforms.

Title translation is an effective and widely used method of creating cross-language links between encyclopedia articles. (Wang et al., 2012; Adafre and de Rijke, 2005) However, title translation alone is not always sufficient. In some cases, for example, the titles of corresponding articles in different languages do not even match. Other methods must be used along with title translation to create a more robust linking tool.

In this paper, we propose a method comprising title and hypernym translation and mixed-language topic model methods to select and link related articles between the English Wikipedia and Baidu Baike online encyclopedias. We also compile a suitable dataset from the above two encyclopedias to evaluate the linking accuracy of our method.

2 Method

Cross-language article linking between different encyclopedias can be formulated as follows: For each encyclopedia K , a collection of human-written articles, can be defined as $K = \{a_i\}_{i=1}^n$, where a_i is an article in K and n is the size of K . Article linking can then be defined as follows: Given two encyclopedia K_1 and K_2 , cross-language article linking is the task of finding the corresponding equivalent article a_j from encyclopedia K_2 for each article a_i from encyclopedia K_1 . Equivalent articles are articles that describe the same topic in different languages.

Our approach to cross-language article linking comprises two stages: candidate selection, which produces a list of candidate articles, and candidate ranking, which ranks that list.

2.1 Candidate Selection

Since knowledge bases (KB) may contain millions of articles, comparison between all possible pairs in two knowledge bases is time-consuming and sometimes impractical. To avoid brute-force comparison, we first select plausible candidate articles on which to focus our efforts. To extract possible candidates, two similarity calculation methods are carried out: title matching and title similarity.

2.1.1 Title Matching

In our title matching method, we formulate candidate selection as an English-Chinese cross-language information retrieval (CLIR) problem (Schönhofen et al., 2008), in which every English article's title is treated as a query and all the articles in the Chinese encyclopedia are treated as the documents. We employ the two main CLIR methods: query translation and document translation.

In query translation, we translate the title of every English article into Chinese and then use these translated titles as queries to retrieve articles from the Chinese encyclopedia. In document translation, we translate the contents of the entire Chinese encyclopedia into English and then search them

using the original English titles. The top 100 results for the query-translation and the top 100 results for document-translation steps are unionized. The resulting list contains our title-matching candidates.

For the query- and document-translation steps, we use the Lucene search engine with similarity scores calculated by the Okapi BM25 ranking function (Beaulieu et al., 1997). We separate all words in the translated and original English article titles with the "OR" operator before submission to the search engine. For all E-C and C-E translation tasks, we use Google Translate.

2.1.2 Title Similarity

In the title similarity method, every Chinese article title is represented as a vector, and each distinct character in all these titles is a dimension of all vectors. The title of each English article is translated into Chinese and represented as a vector. Then, cosine similarity between this vector and the vector of each Chinese title is measured as title similarity.

2.2 Candidate Ranking

The second stage of our approach is to score each viable candidate using a supervised learning method, and then sort all candidates in order of score from high to low as final output.

Each article x_i in KB K_1 can be represented by a feature vector $\mathbf{x}_i = (f_1(x_i), f_2(x_i), \dots, f_n(x_i))$. Also, we have $\mathbf{y}_j = (f_1(y_j), f_2(y_j), \dots, f_n(y_j))$ for a candidate article y_j in KB K_2 . Then, individual feature functions $F_k(x_i, y_j)$ are based on the feature properties of both article a_i and a_j . The top predicted corresponding article y_j in the knowledge base K_2 for an input article x_i in K_1 should receive a higher score than any other entity in $K_2, a_m \in K_2, m \neq j$. We use the support vector machine (SVM) approach to determine the probability of each pair $(\mathbf{x}_i, \mathbf{y}_j)$ being equivalent. Our SVM model's features are described below.

Title Matching and Title Similarity Feature (Baseline)

We use the results of title matching and title similarity from the candidate selection stage as two features for the candidate ranking stage. The similarity values generated by title matching and title similarity are used directly as real value features in the SVM model.

Mixed-language Topic Model Feature (MTM)

For a linked English-Chinese article pair, the distribution of words used in each usually shows some convergence. The two semantically corresponding articles often have many related terms, which results in clusters of specific words. If two articles do not describe the same topic, the distribution of terms is often scattered. (Misra et al., 2008) Thus, the distribution of terms is good measurement of article similarity.

Because the number of all possible words is too large, we adopt a topic model to gather the words into some latent topics. For this feature, we use the Latent Dirichlet Allocation (LDA) (Blei et al., 2003). LDA can be seen as a typical probabilistic approach to latent topic computation. Each topic is represented by a distribution of words, and each word has a probability score used to measure its contribution to the topic. To train the LDA model, the pair English and Chinese articles are concatenated into a single document. English and Chinese terms are all regarded as terms of the same language and the LDA topic model, namely mixed-language topic model, generates both English and Chinese terms for each latent topic. Then, for each English article and Chinese candidate pair in testing, the LDA model provides the distribution of the latent topics. Next, we can use entropy to measure the distribution of topics. The entropy of the estimated topic distribution of a related article is expected to be lower than that of an unrelated article. We can calculate the entropy of the distribution as a value for SVM. The entropy is defined as follows:

$$H = - \sum_{j=1}^T \theta_{dj} \log \theta_{dj}$$

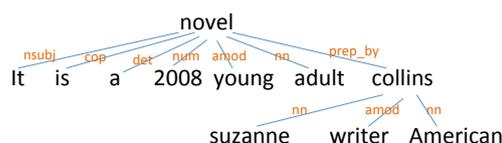
where T is the number of latent topics, θ_{dj} is the topic distribution of a given topic j .

Hypernym Translation Feature (HT)

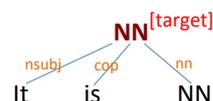
The first sentence of an encyclopedia article usually contains the title of the article. It may also contain a hypernym that defines the category of the article. For example, the first sentence of the “iPad” article in the English Wikipedia begins, “iPad is a line of tablet computers designed and marketed by Apple Inc. . .” In this sentence, the term “tablet computers” is the hypernym of iPad. These extracted hypernyms can be treated as article categories. Therefore, articles containing the same hypernym are likely to belong to the same

category.

In this study, we only carry out title hypernym extraction on the first sentences of English articles due to the looser syntactic structure of Chinese. To generate dependency parse trees for the sentences, we adopt the Stanford Dependency Parser. Then, we manually designed seven patterns to extract hypernyms from the parse tree structures. To demonstrate this idea, let us take the English article “The Hunger Games” for example. The first sentence of this article is “The Hunger Games is a 2008 young adult novel by American writer Suzanne Collins.” Since article titles may be named entities or compound nouns, the dependency parser may mislabel them and thus output an incorrect parse tree. To avoid this problem, we first replace all instances of an article’s title in the first sentence with pronouns. For example, the previous sentence is rewritten as “It is a 2008 young adult novel by American writer Suzanne Collins.” Then, the dependency parser generates the following parse tree:



Next, we apply our predefined syntactic patterns to extract the hypernym. (Hearst, 1992) If any pattern matches the structure of the dependency parse tree, the hypernym can be extracted. In the above example, the following pattern is matched:



In this pattern, the rightmost leaf is the hypernym target. Thus, we can extract the hypernym “novel” from the previous example. The term “novel” is the extracted hypernym of the English article “The Hunger Games”.

After extracting the hypernym of the English article, the hypernym is translated into Chinese. The value of this feature in the SVM model is calculated as follows:

$$F_{hypernym}(h) = \log count(translated(h))$$

where h is the hypernym, $translated(h)$ is the Chinese translation of the term h .

English Title Occurrence Feature (ETO)

In a Baidu Baike article, the first sentence may contain a parenthetical translation of the main title. For example, the first sentence of the Chinese

article on San Francisco is “旧金山（San Francisco），又译‘圣弗朗西斯科’、‘三藩市’”。 We regard the appearance of the English title in the first sentence of a Baidu Baike article as a binary feature: If the English title appears in the first sentence, the value of this feature is 1; otherwise, the value is 0.

3 Evaluation

3.1 Evaluation Dataset

In order to evaluate the performance of cross-language article linking between English Wikipedia and Chinese Baidu Baike, we compile an English-Chinese evaluation dataset from Wikipedia and Baidu Baike online encyclopedias. First, our spider crawls the entire contents of English Wikipedia and Chinese Baidu Baike. Since the two encyclopedias’ article formats differ, we copy the information in each article (title, content, category, etc.) into a standardized XML structure. In order to generate the gold standard evaluation sets of correct English and Chinese article pairs, we automatically collect English-Chinese inter-language links from Wikipedia. For pairs that have both English and Chinese articles, the Chinese article title is regarded as the translation of the English one. Next, we check if there is a Chinese article in Baidu Baike with exactly the same title as the one in Chinese Wikipedia. If so, the corresponding English Wikipedia article and the Baidu Baike article are paired in the gold standard.

To evaluate the performance of our method on linking different types of encyclopedia articles, we compile a set containing the most popular articles. We select the top 500 English-Chinese article pairs with the highest page view counts in Baidu Baike. This set represents the articles people in China are most interested in.

Because our approach uses an SVM model, the data set should be split into training and test sets. For statistical generality, each data set is randomly split 4:1 (training:test) 30 times. The final evaluation results are calculated as the mean of the average of these 30 evaluation sets.

3.2 Evaluation Metrics

To measure the quality of cross-language entity linking, we use the following three metrics. For each English article queries, ten output Baidu Baike candidates are generated in a ranked list. To

define the metrics, we use following notations: N is the number of English query; $r_{i,j}$ is j -th correct Chinese article for i -th English query; $c_{i,k}$ is k -th candidate the system output for i -th English query.

Top- k Accuracy (ACC)

ACC measures the correctness of the first candidate in the candidate list. $ACC = 1$ means that all top candidates are correctly linked (i.e. they match one of the references), and $ACC = 0$ means that none of the top candidates is correct.

$$ACC = \frac{1}{N} \sum_{i=1}^N \left\{ \begin{array}{ll} 1 & \text{if } \exists r_{i,j} : r_{i,j} = c_{i,k} \\ 0 & \text{otherwise} \end{array} \right\}$$

Mean Reciprocal Rank (MRR)

Traditional MRR measures any correct answer produced by the system from among the candidates. $1/MRR$ approximates the average rank of the correct transliteration. An MRR closer to 1 implies that the correct answer usually appears close to the top of the n -best lists.

$$RR_i = \left\{ \begin{array}{ll} \min_j \frac{1}{j} & \text{if } \exists r_{i,j}, c_{i,k} : r_{i,j} = c_{i,k} \\ 0 & \text{otherwise} \end{array} \right\}$$

$$MRR = \frac{1}{N} \sum_{i=1}^N RR_i$$

Recall

Recall is the fraction of the retrieved articles that are relevant to the given query. Recall is used to measure the performance of the candidate selection method. If the candidate selection method can actually select the correct Chinese candidate, the recall will be high.

$$Recall = \frac{|\text{relevant articles}| \cap |\text{retrieved articles}|}{|\text{relevant articles}|}$$

3.3 Evaluation Results

The overall results of our method achieves 80.95% in MRR and 87.46% in recall. Figure 1 shows the top- k ACC from the top 1 to 5. These results show that our method is very effective in linking articles in English Wikipedia to those in Baidu Baike.

In order to show the benefits of each feature used in the SVM model, we conduct an experiment to test the performance of different feature combinations. Because title similarity of the articles is a widely used method, we choose English and Chinese title similarity as the baseline. Then, another feature is added to each configuration until all the features have been added. Table 2 shows the final results of different feature combinations.

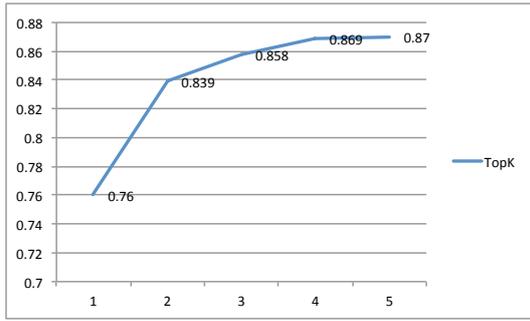


Figure 1: Top- k Accuracy

Level	Configuration	MRR
0	Baseline (BL)	0.6559
1	BL + MTM ^{*1}	0.6967 [†]
	BL + HT ^{*2}	0.6975 [†]
	BL + ETO ^{*3}	0.6981 [†]
2	BL + MTM + HT	0.7703 [†]
	BL + MTM + ETO	0.7558 [†]
	BL + HT + ETO	0.7682 [†]
3	BL + MTM + HT + ETO	0.8095[†]

^{*1}MTM: mix-language topic model

^{*2}HT: hypernym translation

^{*3}ETO: English title occurrence

[†] This config. outperforms the best config. in last level with statistically significant difference.

Table 2: MRRs of Feature Combinations

In the results, we can observe that mix-language topic model, hypernym, and English title occurrence features all noticeably improve the performance. Combining two of these three feature has more improvement and the combination of all the features achieves the best.

4 Discussion

Although our method can effectively generate cross-language links with high accuracy, some correct candidates are not ranked number one. After examining the results, we can divide errors into several categories:

The first kind of error is due to large literal differences between the English and Chinese titles. For example, for the English article “Nero”, our approach ranks the Chinese candidate “尼禄王” (“King Nero”) as number one, instead of the correct answer “尼禄·克劳狄乌斯·德鲁苏斯·日耳曼尼库斯” (the number two candidate). The title of the correct Chinese article is the full name of the Roman Emperor Nero (Nero Claudius Drusus

Germanicus). The false positive “尼禄王” is a historical novel about the life of the Emperor Nero. Because of the large difference in title lengths, the value of the title similarity feature between the English article “Nero” and the corresponding Chinese article is low. Such length differences may cause the SVM model to rank the correct answer lower when the difference of other features are not so significant because the contents of the Chinese candidates are similar.

The second error type is caused by articles that have duplicates in Baidu Baike. For example, for the English article “Jensen Ackles”, our approach generates a link to the Chinese article “Jensen” in Baidu Baike. However, there is another Baidu article “詹森·阿克思” (“Jensen Ackles”). These two articles both describe the actor Jensen Ackles. In this case, our approach still generates a correct link, although it is not the one in the gold standard.

The third error type is translation errors. For example, the English article “Raccoon” is linked to the Baidu article “狸” (raccoon dog), though the correct one is “浣熊” (raccoon). The reason is that Google Translate provides the translation “狸” instead of “浣熊”.

5 Conclusion

Cross-language article linking is the task of creating links between online encyclopedia articles in different languages that describe the same content. We propose a method based on article hypernym and topic model to link English Wikipedia articles to corresponding Chinese Baidu Baike articles. Our method comprises two stages: candidate selection and candidate ranking. We formulate candidate selection as a cross-language information retrieval task based on the title similarity between English and Chinese articles. In candidate ranking, we employ several features of the articles in our SVM model. To evaluate our method, we compile a dataset from English Wikipedia and Baidu Baike, containing the 500 most popular Baidu articles. Evaluation results of our method show an MRR of up to 80.95% and a recall of 87.46%. This shows that our method is effective in generating cross-language links between English Wikipedia and Baidu Baike with high accuracy. Our method does not heavily depend on linguistic characteristics and can be easily extended to generate cross-language article links among different encyclopedias in other languages.

References

- Sisay Fissaha Adafre and Maarten de Rijke. 2005. Discovering missing links in wikipedia. In *Proceedings of the 3rd international workshop on Link discovery (LinkKDD '05)*.
- M. Beaulieu, M. Gatford, X. Huang, S. Robertson, S. Walker, and P. Williams. 1997. Okapi at TREC-5. In *Proceedings of the fifth Text REtrieval Conference (TREC-5)*, pages 143–166.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, volume 2.
- Paul McNamee, James Mayfield, Dawn Lawrie, Douglas W Oard, and David S Doermann. 2011. Cross-language entity linking. In *Proceedings of International Joint Conference on Natural Language Processing (IJCNLP)*, pages 255–263.
- Hemant Misra, Olivier Cappe, and François Yvon. 2008. Using lda to detect semantically incoherent documents. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL '08)*.
- Jong-Hoon Oh, Daisuke Kawahara, Kiyotaka Uchimoto, Jun'ichi Kazama, and Kentaro Torisawa. 2008. Enriching multilingual language resources by discovering missing cross-language links in wikipedia. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, volume 1, pages 322–328.
- Pèter Schönhofen, Andràs Benczùr, István Bìrò, and Kàroly Csalogàny. 2008. Cross-language retrieval with wikipedia. *Advances in Multilingual and Multimodal Information Retrieval, Lecture Notes in Computer Science*, 5152:72–79.
- Philipp Sorg and Philipp Cimiano. 2008. Enriching the crosslingual link structure of wikipedia—a classification-based approach. In *Proceedings of the AAAI 2008 Workshop on Wikipedia and Artificial Intelligence*, pages 49–54.
- Ling-Xiang Tang, In-Su Kang, Fuminori Kimura, Yi-Hsun Lee, Andrew Trotman, Shlomo Geva, and Yue Xu. 2013. Overview of the ntcir-10 cross-lingual link discovery task. In *Proceedings of the Tenth NTCIR Workshop Meeting*.
- Zhichun Wang, Juanzi Li, Zhigang Wang, and Jie Tang. 2012. Cross-lingual knowledge linking across wiki knowledge bases. In *Proceedings of the 21st international conference on World Wide Web (WWW '12)*.

Nonparametric Method for Data-driven Image Captioning

Rebecca Mason and Eugene Charniak

Brown Laboratory for Linguistic Information Processing (BLLIP)

Brown University, Providence, RI 02912

{rebecca, ec}@cs.brown.edu

Abstract

We present a nonparametric density estimation technique for image caption generation. Data-driven matching methods have shown to be effective for a variety of complex problems in Computer Vision. These methods reduce an inference problem for an unknown image to finding an existing labeled image which is semantically similar. However, related approaches for image caption generation (Ordonez et al., 2011; Kuznetsova et al., 2012) are hampered by noisy estimations of visual content and poor alignment between images and human-written captions. Our work addresses this challenge by estimating a word frequency representation of the visual content of a query image. This allows us to cast caption generation as an extractive summarization problem. Our model strongly outperforms two state-of-the-art caption extraction systems according to human judgments of caption relevance.

1 Introduction

Automatic image captioning is a much studied topic in both the Natural Language Processing (NLP) and Computer Vision (CV) areas of research. The task is to identify the visual content of the input image, and to output a relevant natural language caption.

Much prior work treats image captioning as a retrieval problem (see Section 2). These approaches use CV algorithms to retrieve similar images from a large database of captioned images, and then transfer text from the captions of those images to the query image. This is a challenging problem for two main reasons. First, visual similarity measures do not perform reliably and do not

Query Image: Captioned Images:



-
- 1.) 3 month old baby girl with blue eyes in her crib
 - 2.) A photo from the Ismail's **portrait** shoot
 - 3.) A **portrait** of a man, in **black** and **white**
 - 4.) **Portrait** in **black** and **white** with the red rose
 - 5.) I apparently had this saved in **black** and **white** as well
 - 6.) **Portrait** in **black** and **white**
-

Table 1: Example of a query image from the SBU-Flickr dataset (Ordonez et al., 2011), along with scene-based estimates of visually similar images. Our system models visual content using words that are frequent in these captions (highlighted) and extracts a single output caption.

capture all of the relevant details which humans might describe. Second, image captions collected from the web often contain contextual or background information which is not visually relevant to the image being described.

In this paper, we propose a system for transfer-based image captioning which is designed to address these challenges. Instead of selecting an output caption according to a single noisy estimate of visual similarity, our system uses a word frequency model to find a smoothed estimate of visual content across multiple captions, as Table 1 illustrates. It then generates a description of the query image by extracting the caption which best represents the mutually shared content.

The contributions of this paper are as follows:

1. Our caption generation system effectively leverages information from the massive amounts of human-written image captions on the internet. In particular, it exhibits strong performance on the SBU-Flickr dataset (Ordonez et al., 2011), a noisy corpus of one million captioned images collected from the web. We achieve a remarkable 34% improvement in human relevance scores over a recent state-of-the-art image captioning system (Kuznetsova et al., 2012), and 48% improvement over a scene-based retrieval system (Patterson et al., 2014) using the same computed image features.

2. Our approach uses simple models which can be easily reproduced by both CV and NLP researchers. We provide resources to enable comparison against future systems.¹

2 Image Captioning by Transfer

The IM2TEXT model by Ordonez et al. (2011) presents the first web-scale approach to image caption generation. IM2TEXT retrieves the image which is the closest visual match to the query image, and transfers its description to the query image. The COLLECTIVE model by Kuznetsova et al. (2012) is a related approach which uses trained CV recognition systems to detect a variety of visual entities in the query image. A separate description is retrieved for each visual entity, which are then fused into a single output caption. Like IM2TEXT, their approach uses visual similarity as a proxy for textual relevance.

Other related work models the text more directly, but is more restrictive about the source and quality of the human-written training data. Farhadi et al. (2010) and Hodosh et al. (2013) learn joint representations for images and captions, but can only be trained on data with very strong alignment between images and descriptions (i.e. captions written by Mechanical Turkers). Another line of related work (Fan et al., 2010; Aker and Gaizauskas, 2010; Feng and Lapata, 2010) generates captions by extracting sentences from documents which are related to the query image. These approaches are tailored toward specific domains, such as travel and news, where images tend to appear with corresponding text.

¹See http://bllip.cs.brown.edu/download/captioning_resources.zip or ACL Anthology.

3 Dataset

In this paper, we use the SBU-Flickr dataset². Ordonez et al. (2011) query Flickr.com using a huge number of words which describe visual entities, in order to build a corpus of one million images with captions which refer to image content. However, further analysis by Hodosh et al. (2013) shows that many captions in SBU-Flickr (~67%) describe information that cannot be obtained from the image itself, while a substantial fraction (~23%) contain almost no visually relevant information. Nevertheless, this dataset is the only web-scale collection of captioned images, and has enabled notable research in both CV and NLP.³

4 Our Approach

4.1 Overview

For a query image I_q , our task is to generate a relevant description by selecting a single caption from \mathcal{C} , a large dataset of images with human-written captions. In this section, we first define the feature space for visual similarity, then formulate a density estimation problem with the aim of modeling the words which are used to describe visually similar images to I_q . We also explore methods for extractive caption generation.

4.2 Measuring Visual Similarity

Data-driven matching methods have shown to be very effective for a variety of challenging problems (Hays and Efros, 2008; Makadia et al., 2008; Tighe and Lazebnik, 2010). Typically these methods compute global (scene-based) descriptors rather than object and entity detections. Scene-based techniques in CV are generally more robust, and can be computed more efficiently on large datasets.

The basic IM2TEXT model uses an equally weighted average of GIST (Oliva and Torralba, 2001) and TinyImage (Torralba et al., 2008) features, which coarsely localize low-level features in scenes. The output is a multi-dimensional image space where semantically similar scenes (e.g. streets, beaches, highways) are projected near each other.

²<http://tamaraberg.com/CLSP11/>

³In particular, papers stemming from the 2011 JHU-CLSP Summer Workshop (Berg et al., 2012; Dodge et al., 2012; Mitchell et al., 2012) and more recently, the best paper award winner at ICCV (Ordonez et al., 2013).

Patterson and Hays (2012) present “scene attribute” representations which are characterized using low-level perceptual attributes as used by GIST (e.g. openness, ruggedness, naturalness), as well as high-level attributes informed by open-ended crowd-sourced image descriptions (e.g., indoor lighting, running water, places for learning). Follow-up work (Patterson et al., 2014) shows that their attributes provide improved matching for image captioning over IM2TEXT baseline. We use their publicly available⁴ scene attributes for our experiments. Training set and query images are represented using 102-dimensional real-valued vectors, and similarity between images is measured using the Euclidean distance.

4.3 Density Estimation

As shown in Bishop (2006), probability density estimates at a particular point can be obtained by considering points in the training data within some local neighborhood. In our case, we define some region \mathcal{R} in the image space which contains I_q . The probability mass of that space is

$$P = \int_{\mathcal{R}} p(I_q) dI_q \quad (1)$$

and if we assume that \mathcal{R} is small enough such that $p(I_q)$ is roughly constant in \mathcal{R} , we can approximate

$$p(I_q) \approx \frac{k^{img}}{n^{img} V^{img}} \quad (2)$$

where k^{img} is the number of images within \mathcal{R} in the training data, n^{img} is the total number of images in the training data, and V^{img} is the volume of \mathcal{R} . In this paper, we fix k^{img} to a constant value, so that V^{img} is determined by the training data around the query image.⁵

At this point, we extend the density estimation technique in order to estimate a smoothed model of descriptive text. Let us begin by considering $p(w|I_q)$, the conditional probability of the word⁶ w given I_q . This can be described using a

⁴https://github.com/genp/sun_attributes

⁵As an alternate approach, one could fix the value of V^{img} and determine k^{img} from the number of points in \mathcal{R} , giving rise to the kernel density approach (a.k.a. *Parzen windows*). However we believe the KNN approach is more appropriate here, because the number of samples is nearly 10000 times greater than the number of dimensions in the image representation.

⁶Here, we use word to refer to non-function words, and assume all function words have been removed from the captions.

Bayesian model:

$$p(w|I_q) = \frac{p(I_q|w)p(w)}{p(I_q)} \quad (3)$$

The prior for w is simply its unigram frequency in \mathcal{C} , where n_w^{txt} and n^{txt} are word token counts:

$$p(w) = \frac{n_w^{txt}}{n^{txt}} \quad (4)$$

Note that n^{txt} is not the same as n^{img} because a single captioned image can have multiple words in its caption. Likewise, the conditional density

$$p(I_q|w) \approx \frac{k_w^{txt}}{n_w^{txt} V^{img}} \quad (5)$$

considers instances of observed words within \mathcal{R} , although the volume of \mathcal{R} is still defined by the image space. k_w^{txt} is the number of times w is used within \mathcal{R} while n_w^{txt} is the total number of times w is observed in \mathcal{C} .

Combining Equations 2, 4, and 5 and canceling out terms gives us the posterior probability:

$$p(w|I_q) = \frac{k_w^{txt}}{k^{img}} \cdot \frac{n^{img}}{n^{txt}} \quad (6)$$

If the number of words in each caption is independent of its image’s location in the image space, then $p(w|I_q)$ is approximately the observed unigram frequency for the captions inside \mathcal{R} .

4.4 Extractive Caption Generation

We compare two selection methods for extractive caption generation:

1. SumBasic SumBasic (Nenkova and Vanderwende, 2005) is a sentence selection algorithm for extractive multi-document summarization which exclusively maximizes the appearance of words which have high frequency in the original documents. Here, we adapt SumBasic to maximize the average value of $p(w|I_q)$ in a single extracted caption:

$$output = \arg \max_{c^{txt} \in \mathcal{R}} \sum_{w \in c^{txt}} \frac{1}{|c^{txt}|} p(w|I_q) \quad (7)$$

The candidate captions c^{txt} do not necessarily have to be observed in \mathcal{R} , but in practice we did not find increasing the number of candidate captions to be more effective than increasing the size of \mathcal{R} directly.

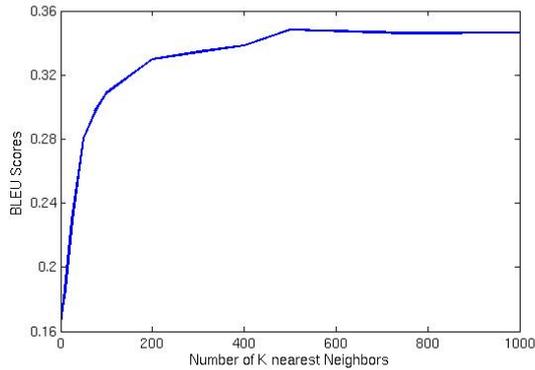


Figure 1: BLEU scores vs k for SumBasic extraction.

2. KL Divergence We also consider a KL Divergence selection method. This method outperforms the SumBasic selection method for extractive multi-document summarization (Haghighi and Vanderwende, 2009). It also generates the best extractive captions for Feng and Lapata (2010), who caption images by extracting text from a related news article. The KL Divergence method is

$$output = \arg \min_{c^{txt} \in \mathcal{R}} \sum_w p(w|I_q) \log \frac{p(w|I_q)}{p(w|c^{txt})} \quad (8)$$

5 Evaluation

5.1 Automatic Evaluation

Although BLEU (Papineni et al., 2002) scores are widely used for image caption evaluation, we find them to be poor indicators of the quality of our model. As shown in Figure 1, our system’s BLEU scores increase rapidly until about $k = 25$. Past this point we observe the density estimation seems to get washed out by oversmoothing, but the BLEU scores continue to improve until $k = 500$ but only because the generated captions become increasingly shorter. Furthermore, although we observe that our SumBasic extracted captions obtain consistently higher BLEU scores, our personal observations find KL Divergence captions to be better at balancing recall and precision. Nevertheless, BLEU scores are the accepted metric for recent work, and our KL Divergence captions with $k = 25$ still outperform all other previously published systems and baselines. We omit full results here due to space, but make our BLEU setup with captions for all systems and baselines available for documentary purposes.

System	Relevance
COLLECTIVE	2.38 ($\sigma = 1.45$)
SCENE ATTRIBUTES	2.15 ($\sigma = 1.45$)
SYSTEM	3.19 ($\sigma = 1.50$)
HUMAN	4.09 ($\sigma = 1.14$)

Table 2: Human evaluations of relevance: mean ratings and standard deviations. See Section 5.2.

5.2 Human Evaluation

We perform our human evaluation of caption relevance using a similar setup to that of Kuznetsova et al. (2012), who have humans rate the image captions on a 1-5 scale (5: perfect, 4: almost perfect, 3: 70-80% good, 2: 50-70% good, 1: totally bad). Evaluation is performed using Amazon Mechanical Turk. Evaluators are shown both the caption and the query image, and are specifically instructed to ignore errors in grammaticality and coherence.

We generate captions using our system with KL Divergence sentence selection and $k = 25$. We also evaluate the original HUMAN captions for the query image, as well as generated captions from two recently published caption transfer systems. First, we consider the SCENE ATTRIBUTES system (Patterson et al., 2014), which represents both the best scene-based transfer model and a $k = 1$ nearest-neighbor baseline for our system. We also compare against the COLLECTIVE system (Kuznetsova et al., 2012), which is the best object-based transfer model.

In order to facilitate comparison, we use the same test/train split that is used in the publicly available system output for the COLLECTIVE system⁷. However, we remove some query images which have contamination between the train and test set (this occurs when a photographer takes multiple shots of the same scene and gives all the images the exact same caption). We also note that their test set is selected based on images where their object detection systems had good performance, and may not be indicative of their performance on other query images.

Table 2 shows the results of our human study. Captions generated by our system have 48% improvement in relevance over the SCENE ATTRIBUTES system captions, and 34% improve-

⁷<http://www.cs.sunysb.edu/~pkuznetsova/generation/cogn/captions.html>

				
COLLECTIVE:	One of the birds seen in company of female and juvenile.	View of this woman sitting on the sidewalk in Mumbai by the stained glass. The boy walking by next to matching color walls in gov t building.	Found this mother bird feeding her babies in our maple tree on the phone.	Found in floating grass spotted alongside the scenic North Cascades Hwy near Ruby arm a black bear.
SCENE ATTRIBUTES:	This small bird is pretty much only found in the ancient Caledonian pine forests of the Scottish Highlands.	me and allison in front of the white house	The sand in this beach was black...I repeat BLACK SAND	Not the green one, but the almost ghost-like white one in front of it.
SYSTEM:	White bird found in park standing on brick wall	by the white house	pine tree covered in ice :)	Pink flower in garden w/ moth
HUMAN:	Some black head bird taken in bray head.	Us girls in front of the white house	Male cardinal in snowy tree knots	Black bear by the road between Ucluelet and Port Alberni, B.C., Canada

Table 3: Example query images and generated captions.

ment over the COLLECTIVE system captions. Although our system captions score lower than the human captions on average, there are some instances of our system captions being judged as more relevant than the human-written captions.

6 Discussion and Examples

Example captions are shown in Table 3. In many instances, scene-based image descriptors provide enough information to generate a complete description of the image, or at least a sufficiently good one. However, there are some kinds of images for which scene-based features alone are insufficient. For example, the last example describes the small pink flowers in the background, but misses the bear.

Image captioning is a relatively novel task for which the most compelling applications are probably not yet known. Much previous work in image captioning focuses on generating captions that concretely describe detected objects and entities (Kulkarni et al., 2011; Yang et al., 2011; Mitchell et al., 2012; Yu and Siskind, 2013). However, human-generated captions and annotations also describe perceptual features, contextual information, and other types of content. Additionally, our system is robust to instances where entity detection systems fail to perform. However, one could

consider combined approaches which incorporate more regional content structures. For example, previous work in nonparametric hierarchical topic modeling (Blei et al., 2010) and scene labeling (Liu et al., 2011) may provide avenues for further improvement of this model. Compression methods for removing visually irrelevant information (Kuznetsova et al., 2013) may also help increase the relevance of extracted captions. We leave these ideas for future work.

References

- Ahmet Aker and Robert Gaizauskas. 2010. Generating image descriptions using dependency relational patterns. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, pages 1250–1258, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Alexander C Berg, Tamara L Berg, Hal Daume, Jesse Dodge, Amit Goyal, Xufeng Han, Alyssa Mensch, Margaret Mitchell, Aneesh Sood, Karl Stratos, et al. 2012. Understanding and predicting importance in images. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3562–3569. IEEE.
- Christopher M Bishop. 2006. *Pattern recognition and machine learning*, volume 1. Springer New York.
- David M. Blei, Thomas L. Griffiths, and Michael I. Jordan. 2010. The nested chinese restaurant process

- and bayesian nonparametric inference of topic hierarchies. *J. ACM*, 57(2):7:1–7:30, February.
- Jesse Dodge, Amit Goyal, Xufeng Han, Alyssa Mensch, Margaret Mitchell, Karl Stratos, Kota Yamaguchi, Yejin Choi, Hal Daumé III, Alexander C. Berg, and Tamara L. Berg. 2012. Detecting visual text. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.
- Xin Fan, Ahmet Aker, Martin Tomko, Philip Smart, Mark Sanderson, and Robert Gaizauskas. 2010. Automatic image captioning from the web for gps photographs. In *Proceedings of the international conference on Multimedia information retrieval*, MIR '10, pages 445–448, New York, NY, USA. ACM.
- Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. 2010. Every picture tells a story: generating sentences from images. In *Proceedings of the 11th European conference on Computer vision: Part IV, ECCV'10*, pages 15–29, Berlin, Heidelberg. Springer-Verlag.
- Yansong Feng and Mirella Lapata. 2010. How many words is a picture worth? automatic caption generation for news images. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1239–1249, Stroudsburg, PA, USA. 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. Association for Computational Linguistics.
- James Hays and Alexei A Efros. 2008. Im2gps: estimating geographic information from a single image. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.
- Micah Hodosh, Peter Young, and Julia Hockenmaier. 2013. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899.
- Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C. Berg, and Tamara L. Berg. 2011. Baby talk: Understanding and generating simple image descriptions. In *CVPR*, pages 1601–1608.
- Polina Kuznetsova, Vicente Ordonez, Alexander C. Berg, Tamara L. Berg, and Yejin Choi. 2012. Collective generation of natural image descriptions. In *ACL*.
- Polina Kuznetsova, Vicente Ordonez, Alexander Berg, Tamara Berg, and Yejin Choi. 2013. Generalizing image captions for image-text parallel corpus. In *ACL*.
- Ce Liu, Jenny Yuen, and Antonio Torralba. 2011. Nonparametric scene parsing via label transfer. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(12):2368–2382.
- Ameesh Makadia, Vladimir Pavlovic, and Sanjiv Kumar. 2008. A new baseline for image annotation. In *Computer Vision–ECCV 2008*, pages 316–329. Springer.
- Margaret Mitchell, Jesse Dodge, Amit Goyal, Kota Yamaguchi, Karl Stratos, Xufeng Han, Alyssa Mensch, Alexander C. Berg, Tamara L. Berg, and Hal Daumé III. 2012. Midge: Generating image descriptions from computer vision detections. In *European Chapter of the Association for Computational Linguistics (EACL)*.
- Ani Nenkova and Lucy Vanderwende. 2005. The impact of frequency on summarization.
- Aude Oliva and Antonio Torralba. 2001. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42:145–175.
- V. Ordonez, G. Kulkarni, and T.L. Berg. 2011. Im2text: Describing images using 1 million captioned photographs. In *NIPS*.
- Vicente Ordonez, Jia Deng, Yejin Choi, Alexander C Berg, and Tamara L Berg. 2013. From large scale image categorization to entry-level categories. In *International Conference on Computer Vision*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Weijing 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.
- Genevieve Patterson and James Hays. 2012. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2751–2758. IEEE.
- Genevieve Patterson, Chen Xu, Hang Su, and James Hays. 2014. The sun attribute database: Beyond categories for deeper scene understanding. *International Journal of Computer Vision*.
- Joseph Tighe and Svetlana Lazebnik. 2010. Superparsing: scalable nonparametric image parsing with superpixels. In *Computer Vision–ECCV 2010*, pages 352–365. Springer.
- Antonio Torralba, Robert Fergus, and William T Freeman. 2008. 80 million tiny images: A large data set for nonparametric object and scene recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(11):1958–1970.

Yezhou Yang, Ching Lik Teo, Hal Daumé III, and Yian-nis Aloimonos. 2011. Corpus-guided sentence generation of natural images. In *Empirical Methods in Natural Language Processing (EMNLP)*, Edinburgh, Scotland.

Haonan Yu and Jeffrey Mark Siskind. 2013. Grounded language learning from video described with sentences. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 53–63, Sofia, Bulgaria. Association for Computational Linguistics.

Improved Correction Detection in Revised ESL Sentences

Huichao Xue and Rebecca Hwa

Department of Computer Science,
University of Pittsburgh,

210 S Bouquet St, Pittsburgh, PA 15260, USA

{hux10, hwa}@cs.pitt.edu

Abstract

This work explores methods of automatically detecting corrections of individual mistakes in sentence revisions for ESL students. We have trained a classifier that specializes in determining whether consecutive basic-edits (word insertions, deletions, substitutions) address the same mistake. Experimental result shows that the proposed system achieves an F_1 -score of 81% on correction detection and 66% for the overall system, out-performing the baseline by a large margin.

1 Introduction

Quality feedback from language tutors can help English-as-a-Second-Language (ESL) students improve their writing skills. One of the tutors' tasks is to isolate writing mistakes within sentences, and point out (1) why each case is considered a mistake, and (2) how each mistake should be corrected. Because this is time consuming, tutors often just rewrite the sentences without giving any explanations (Fregeau, 1999). Due to the effort involved in comparing revisions with the original texts, students often fail to learn from these revisions (Williams, 2003).

Computer aided language learning tools offer a solution for providing more detailed feedback. Programs can be developed to compare the student's original sentences with the tutor-revised sentences. Swanson and Yamangil (2012) have proposed a promising framework for this purpose. Their approach has two components: one to detect individual corrections within a revision, which they termed *correction detection*; another to determine what the correction fixes, which they termed *error type selection*. Although they reported a high accuracy for the error type selection classifier alone, the bottleneck of their system is the other

component – correction detection. An analysis of their system shows that approximately 70% of the system's mistakes are caused by mis-detections in the first place. Their correction detection algorithm relies on a set of heuristics developed from one single data collection (the FCE corpus (Yannakoudakis et al., 2011)). When determining whether a set of basic-edits (word insertions, deletions, substitutions) contributes to the same correction, these heuristics lack the flexibility to adapt to a specific context. Furthermore, it is not clear if the heuristics will work as well for tutors trained to mark up revisions under different guidelines.

We propose to improve upon the correction detection component by training a classifier that determines which edits in a revised sentence address the same error in the original sentence. The classifier can make more accurate decisions adjusted to contexts. Because the classifier were trained on revisions where corrections are explicitly marked by English experts, it is also possible to build systems adjusted to different annotation standards.

The contributions of this paper are: (1) We show empirically that a major challenge in correction detection is to determine the number of edits that address the same error. (2) We have developed a merging model that reduces mis-detection by 1/3, leading to significant improvement in the accuracies of combined *correction detection* and *error type selection*. (3) We have conducted experiments across multiple corpora, indicating that the proposed merging model is generalizable.

2 Correction Detection

Comparing a student-written sentence with its revision, we observe that each correction can be decomposed into a set of more basic edits such as word insertions, word deletions and word substitutions. In the example shown in Figure 1, the correction “*to change* \Rightarrow *changing*” is composed of a deletion of *to* and a substitution from *change*

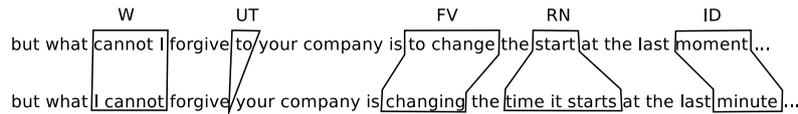


Figure 1: Detecting corrections from revisions. Our system detects individual corrections by comparing the original sentence with its revision, so that each correction addresses one error. Each polygon corresponds to one correction; the labels are codes of the error types. The codes follow the annotation standard in FCE corpus (Nicholls, 2003). In this example, *W* is incorrect Word order; *UT* is Unnecessary preposition; *FV* is wrong Verb Form; *RN* is Nnoun needs to be Replaced; *ID* is IDiom error.

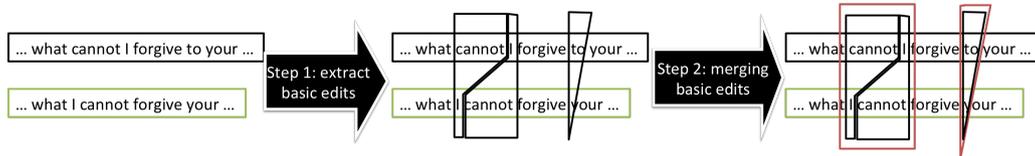


Figure 2: A portion of the example from Figure 1 undergoing the two-step correction detection process. The basic edits are indicated by black polygons. The corrections are shown in red polygons.

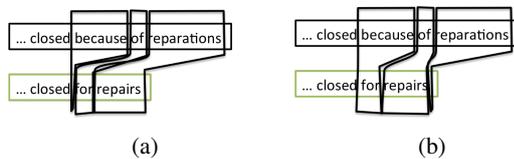
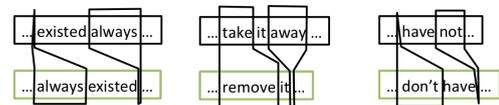


Figure 3: Basic edits extracted by the edit-distance algorithm (Levenshtein, 1966) do not necessarily match our linguistic intuition. The ideal basic-edits are shown in Figure 3a, but since the algorithm only cares about minimizing the number of edits, it may end up extracting basic-edits shown in Figure 3b.

to *changing*; the correction “*moment* \Rightarrow *minute*” is itself a single word substitution. Thus, we can build systems to detect corrections which operates in two steps: (1) detecting the basic edits that took place during the revision, and (2) merging those basic edits that address the same error. Figure 2 illustrates the process for a fragment of the example sentence from Figure 1.

In practice, however, this two-step approach may result in mis-detections due to ambiguities. Mis-detections may be introduced from either steps. While detecting basic edits, Figure 3 gives an example of problems that might arise. Because the Levenshtein algorithm only tries to minimize the number of edits, it does not care whether the edits make any linguistic sense. For merging basic edits, Swanson and Yamangil applied a distance heuristic – basic-edits that are close to each other (e.g. basic edits with at most one word lying in between) are merged. Figure 4 shows cases for which the heuristic results in the wrong scope.

These errors caused their system to mis-detect 30% of the corrections. Since mis-detected corrections cannot be analyzed down the pipeline,



(a) The basic edits are addressing the same problem. But these basic edits are non-adjacent, and therefore not merged by S&Y’s algorithm.

(b) The basic edits in the above two cases address different problems though they are adjacent. S&Y’s merging algorithm incorrectly merges them.

Figure 4: Merging mistakes by the algorithm proposed in Swanson and Yamangil (2012) (S&Y), which merges adjacent basic edits.

the correction detection component became the bottle-neck of their overall system. Out of the 42% corrections that are incorrectly analyzed¹, 30%/42% \approx 70% are caused by mis-detections in the first place. An improvement in correction detection may increase the system accuracy overall.

We conducted an error analysis to attribute errors to either step when the system detects a wrong set of corrections for a sentence. We examine the first step’s output. If the resulting basic edits do not match with those that compose the actual corrections, we attribute the error to the first step. Otherwise, we attribute the error to the second step. Our analysis confirms that the merging step is the bottleneck in the current correction detection system – it accounts for 75% of the mis-detections. Therefore, to effectively reduce the algorithm’s mis-detection errors, we propose to

¹Swanson and Yamangil reported an overall system with 58% F-score.

build a classifier to merge with better accuracies.

Other previous tasks also involve comparing two sentences. Unlike evaluating grammar error correction systems (Dahlmeier and Ng, 2012), correction detection cannot refer to a gold standard. Our error analysis above also highlights our task’s difference with previous work that identify corresponding phrases between two sentences, including phrase extraction (Koehn et al., 2003) and paraphrase extraction (Cohn et al., 2008). They are fundamentally different in that the granularity of the extracted phrase pairs is a major concern in our work – we need to guarantee each detected phrase pair to address exactly one writing problem. In comparison, phrase extraction systems aim to improve the end-to-end MT or paraphrasing systems. A bigger concern is to guarantee the extracted phrase pairs are indeed translations or paraphrases. Recent work therefore focuses on identifying the alignment/edits between two sentences (Snover et al., 2009; Heilman and Smith, 2010).

3 A Classifier for Merging Basic-Edits

Figure 4 highlights the problems with indiscriminantly merging basic-edits that are adjacent. Intuitively, it seems that the decision should be more context dependent. Certain patterns may indicate that two adjacent basic-edits are a part of the same correction while others may indicate that they each address a different problem. For example, in Figure 5a, when the insertion of one word is followed by the deletion of the same word, the insertion and deletion are likely addressing one single error. This is because these two edits would combine together as a word-order change. On the other hand, in Figure 5b, if one edit includes a substitution between words with the same POS’s, then it is likely fixing a word choice error by itself. In this case, it should not be merged with other edits.

To predict whether two basic-edits address the same writing problem more discriminatively, we train a Maximum Entropy binary classifier based on features extracted from relevant contexts for the basic edits. We use features in Table 1 in the proposed classifier. We design the features to indicate: **(A)** whether merging the two basic-edits matches the pattern for a common correction. **(B)** whether one basic-edit addresses one single error.

We train the classifier using samples extracted from revisions where individual corrections are explicitly annotated. We first extract the basic-



(a) The pattern indicates that the two edits address the same problem
 (b) The pattern indicates that the two edits do not address the same problem

Figure 5: Patterns indicating whether two edits address the same writing mistake.

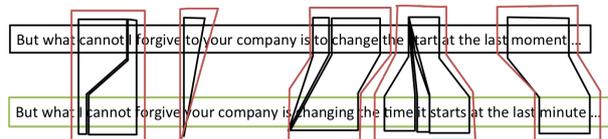


Figure 6: Extracting training instances for the merger. Our goal is to train classifiers to tell if two basic edits should be merged (*True* or *False*). We break each correction (outer polygons, also colored in red) in the training corpus into a set of basic edits (black polygons). We construct an instance for each consecutive pair of basic edits. If two basic edits were extracted from the same correction, we will mark the outcome as *True*, otherwise we will mark the outcome as *False*.

edits that compose each correction. We then create a training instance for each pair of two consecutive basic edits: if two consecutive basic edits need to be merged, we will mark the outcome as *True*, otherwise it is *False*. We illustrate this in Figure 6.

4 Experimental Setup

We combine Levenshtein algorithm with different merging algorithms for correction detection.

4.1 Dataset

An ideal data resource would be a real-world collection of student essays and their revisions (Tajiri et al., 2012). However, existing revision corpora do not have the fine-grained annotations necessary for our experimental gold standard. We instead use error annotated data, in which the corrections were provided by human experts. We simulate the revisions by applying corrections onto the original sentence. The teachers’ annotations are treated as gold standard for the detailed corrections.

We considered four corpora with different ESL populations and annotation standards, including FCE corpus (Yannakoudakis et al., 2011), NU-CLE corpus (Dahlmeier et al., 2013), UIUC corpus² (Rozovskaya and Roth, 2010) and HOO2011 corpus (Dale and Kilgarriff, 2011). These corpora all provide experts’ corrections along with error

²UIUC corpus contains annotations of essays collected from ICLE (Granger, 2003) and CLEC (Gui and Yang, 2003).

Type	name	description
A	gap-between-edits	Gap between the two edits. In particular, we use the number of words between the two edits' original words, as well as the revised words. Note that Swanson and Yamangil's approach is a special case that only considers if the basic-edits have zero gap in both sentences.
	tense-change	We detect patterns such as: if the original-revision pair matches the pattern "V-ing \Rightarrow to V".
	word-order-error	Whether the basic-edits' original word set and the revised word set are the same (one or zero).
	same-word-set	If the original sentence and the revised sentence have the same word set, then it's likely that all the edits are fixing the word order error.
	revised-to	The phrase comprised of the two revised words.
B	editdistance=1	If one basic-edit is a substitution, and the original/revised word only has 1 edit distance, it indicates that the basic-edit is fixing a misspelling error.
	not-in-dict	If the original word does not have a valid dictionary entry, then it indicates a misspelling error.
	word-choice	If the original and the revised words have the same POS, then it is likely fixing a word choice error.
	preposition-error	Whether the original and the revised words are both prepositions.

Table 1: Features used in our proposed classifier.

corpus	sentences	sentences with ≥ 2 corrections revised sentences
FCE	33,900	53.45%
NUCLE	61,625	48.74%
UIUC	883	61.32%
HOO2011	966	42.05%

Table 2: Basic statistics of the corpora that we consider.

type mark-ups. The basic statistics of the corpora are shown in Table 2. In these corpora, around half of revised sentences contains multiple corrections. We have split each corpus into 11 equal parts. One part is used as the development dataset; the rest are used for 10-fold cross validation.

4.2 Evaluation Metrics

In addition to evaluating the merging algorithms on the stand-alone task of correction detection, we have also plugged in the merging algorithms into an end-to-end system in which every automatically detected correction is further classified into an error type. We replicated the error type selector described in Swanson and Yamangil (2012). The error type selector's accuracies are shown in Table 3³. We compare two merging algorithms, combined with Levenshtein algorithm:

S&Y The merging heuristic proposed by Swanson and Yamangil, which merges the adjacent basic edits into single corrections.

MaxEntMerger We use the Maximum Entropy classifier to predict whether we should merge the two edits, as described in Section 3⁴.

We evaluate extrinsically the merging components' effect on overall system performance by

³Our replication has a slightly lower error type selection accuracy on FCE (80.02%) than the figure reported by Swanson and Yamangil (82.5%). This small difference on error type selection does not affect our conclusions about correc-

Corpus	Error Types	Accuracy
FCE	73	80.02%
NUCLE	27	67.36%
UIUC	8	80.23%
HOO2011	38	64.88%

Table 3: Error type selection accuracies on different corpora. We use a Maximum Entropy classifier along with features suggested by Swanson and Yamangil for this task. The reported figures come from 10-fold cross validations on different corpora.

comparing the boundaries of system's detected corrections with the gold standard. We evaluate both (1) the F-score in detecting corrections (2) the F-score in correctly detecting both the corrections' and the error types they address.

5 Experiments

We design experiments to answer two questions:

1. Do the additional contextual information about correction patterns help guide the merging decisions? How much does a classifier trained for this task improve the system's overall accuracy?
2. How well does our method generalize over revisions from different sources?

Our major experimental results are presented in Table 4 and Table 6. Table 4 compares the overall educational system's accuracies with different merging algorithms. Table 6 shows the system's F_1 score when trained and tested on different corpora. We make the following observations:

First, Table 4 shows that by incorporating correction patterns into the merging algorithm, the

tion detection.

⁴We use the implementation at http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html.

errors in correction detection step were reduced. This led to a significant improvement on the overall system’s F_1 -score on all corpora. The improvement is most noticeable on FCE corpus, where the error in correction detection step was reduced by 9%. That is, one third of the correction mis-detections were eliminated. Table 5 shows that the number of merging errors are significantly reduced by the new merging algorithm. In particular, the number of false positives (system proposes merges when it should not) is significantly reduced.

Second, our proposed model is able to generalize over different corpora. As shown in Table 6. The models built on corpora can generally improve the correction detection accuracy⁵. Models built on the same corpus generally perform the best. Also, as suggested by the experimental result, among the four corpora, FCE corpus is a comparably good resource for training correction detection models with our current feature set. One reason is that FCE corpus has many more training instances, which benefits model training. We tried varying the training dataset size, and test it on different corpora. Figure 7 suggests that the model’s accuracies increase with the training corpus size.

6 Conclusions

A revision often contains multiple corrections that address different writing mistakes. We explore building computer programs to accurately detect individual corrections in one single revision. One major challenge lies in determining whether consecutive basic-edits address the same mistake. We propose a classifier specialized in this task. Our experiments suggest that: (1) the proposed classifier reduces correction mis-detections in previous systems by 1/3, leading to significant overall system performance. (2) our method is generalizable over different data collections.

Acknowledgements

This work is supported by U.S. National Science Foundation Grant IIS-0745914. We thank the anonymous reviewers for their suggestions; we also thank Homa Hashemi, Wencan Luo, Fan Zhang, Lingjia Deng, Wenting Xiong and Yafei Wei for helpful discussions.

⁵We currently do not evaluate the end-to-end system over different corpora. This is because different corpora employ different error type categorization standards.

Method	Corpus	Correction Detection F_1	Overall F_1 -score
S&Y	FCE	70.40%	57.10%
MaxEntMerger	FCE	80.96%	66.36%
S&Y	NUCLE	61.18%	39.32%
MaxEntMerger	NUCLE	63.88%	41.00%
S&Y	UIUC	76.57%	65.08%
MaxEntMerger	UIUC	82.81%	70.55%
S&Y	HOO2011	68.73%	50.95%
MaxEntMerger	HOO2011	75.71%	56.14%

Table 4: Extrinsic evaluation, where we plugged the two merging models into an end-to-end feedback detection system by Swanson and Yamangil.

Merging algorithm	TP	FP	FN	TN
S&Y	33.73%	13.46%	5.71%	47.10%
MaxEntMerger	36.04%	3.26%	3.41%	57.30%

Table 5: Intrinsic evaluation, where we evaluate the proposed merging model’s prediction accuracy on FCE corpus. This table shows a breakdown of true-positives (TP), false-positives (FP), false-negatives (FN) and true-negatives (TN) for the system built on FCE corpus.

training \ testing	FCE	NUCLE	UIUC	HOO2011
S&Y	70.44	61.18%	76.57%	68.73%
FCE	80.96%	61.26%	83.07%	75.43%
NUCLE	74.53%	63.88%	78.57%	74.73%
UIUC	77.25%	58.21%	82.81%	70.83%
HOO2011	71.94%	54.99%	71.19%	75.71%

Table 6: Correction detection experiments by building the model on one corpus, and applying it onto another. We evaluate the correction detection performance with F_1 score. When training and testing on the same corpus, we run a 10-fold cross validation.

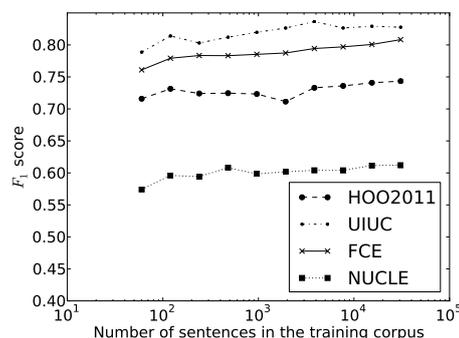


Figure 7: We illustrate the performance of correction detection systems trained on subsets of FCE corpus. Each curve in this figure represents the F_1 -scores for correction detection of the model trained on a subset of FCE and tested on different corpora. When testing on FCE, we used $\frac{1}{11}$ of the FCE corpus, which we kept as development data.

References

- Trevor Cohn, Chris Callison-Burch, and Mirella Lapata. 2008. Constructing corpora for the development and evaluation of paraphrase systems. *Computational Linguistics*, 34(4):597–614.
- Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572, Montréal, Canada, June. Association for Computational Linguistics.
- 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 *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31.
- Robert Dale and Adam Kilgarriff. 2011. Helping our own: The HOO 2011 pilot shared task. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 242–249. Association for Computational Linguistics.
- Laureen A Fregeau. 1999. Preparing ESL students for college writing: Two case studies. *The Internet TESL Journal*, 5(10).
- Sylviane Granger. 2003. The International Corpus of Learner English: a new resource for foreign language learning and teaching and second language acquisition research. *Tesol Quarterly*, 37(3):538–546.
- Shicun Gui and Huizhong Yang. 2003. Zhongguo xuexizhe yingyu yuliaohu.(chinese learner english corpus). *Shanghai: Shanghai Waiyu Jiaoyu Chubanshe*.
- Michael Heilman and Noah A Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019. Association for Computational Linguistics.
- P. Koehn, F.J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics.
- V. I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707710.
- D. Nicholls. 2003. The Cambridge Learner Corpus: Error coding and analysis for lexicography and ELT. In *Proceedings of the Corpus Linguistics 2003 conference*, pages 572–581.
- Alla Rozovskaya and Dan Roth. 2010. Annotating ESL errors: Challenges and rewards. In *Proceedings of the NAACL HLT 2010 fifth workshop on innovative use of NLP for building educational applications*, pages 28–36. Association for Computational Linguistics.
- Matthew G Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. 2009. TER-Plus: paraphrase, semantic, and alignment enhancements to translation edit rate. *Machine Translation*, 23(2-3):117–127.
- Ben Swanson and Elif Yamangil. 2012. Correction detection and error type selection as an ESL educational aid. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 357–361, Montréal, Canada, June. Association for Computational Linguistics.
- Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for ESL learners using global context. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 198–202. Association for Computational Linguistics.
- Jason Gordon Williams. 2003. Providing feedback on ESL students written assignments. *The Internet TESL Journal*, 4(10).
- Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 180–189. Association for Computational Linguistics.

Unsupervised Alignment of Privacy Policies using Hidden Markov Models

Rohan Ramanath Fei Liu Norman Sadeh Noah A. Smith

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA

{rrohan, feiliu, sadeh, nasmith}@cs.cmu.edu

Abstract

To support empirical study of online privacy policies, as well as tools for users with privacy concerns, we consider the problem of aligning sections of a thousand policy documents, based on the issues they address. We apply an unsupervised HMM; in two new (and reusable) evaluations, we find the approach more effective than clustering and topic models.

1 Introduction

Privacy policy documents are verbose, often esoteric legal documents that many people encounter as clients of companies that provide services on the web. McDonald and Cranor (2008) showed that, if users were to read the privacy policies of every website they access during the course of a year, they would end up spending a substantial amount of their time doing just that and would often still not be able to answer basic questions about what these policies really say. Unsurprisingly, many people do not read them (Federal Trade Commission, 2012).

Such policies therefore offer an excellent opportunity for NLP tools that summarize or extract key information that (i) helps users understand the implications of agreeing to these policies and (ii) helps legal analysts understand the contents of these policies and make recommendations on how they can be improved or made more clear. Past applications of NLP have sought to parse privacy policies into machine-readable representations (Brodie et al., 2006) or extract sub-policies from larger documents (Xiao et al., 2012). Machine learning has been applied to assess certain attributes of policies (Costante et al., 2012; Ammar et al., 2012; Costante et al., 2013; Zimbeck and Bellovin, 2013).

This paper instead analyzes policies in aggregate, seeking to *align* sections of policies. This

task is motivated by an expectation that many policies will address similar issues,¹ such as collection of a user’s contact, location, health, and financial information, sharing with third parties, and deletion of data. This expectation is supported by recommendation by privacy experts (Gellman, 2014) and policymakers (Federal Trade Commission, 2012); in the financial services sector, the Gramm-Leach-Bliley Act *requires* these institutions to address a specific set of issues. Aligning policy sections is a first step toward our aforementioned summarization and extraction goals.

We present the following contributions:

- A new corpus of over 1,000 privacy policies gathered from widely used websites, manually segmented into subtitled sections by crowdworkers (§2).
- An unsupervised approach to aligning the policy sections based on the issues they discuss. For example, sections that discuss “user data on the company’s server” should be grouped together. The approach is inspired by the application of hidden Markov models to sequence alignment in computational biology (Durbin et al., 1998; §3).
- Two reusable evaluation benchmarks for the resulting alignment of policy sections (§4). We demonstrate that our approach outperforms naïve methods (§5).

Our corpus and benchmarks are available at <http://usableprivacy.org/data>.

2 Data Collection

We collected 1,010 unique privacy policy documents from the top websites ranked by Alexa.com.² These policies were collected during a period of six weeks during December 2013 and January 2014. They are a snapshot of privacy policies of mainstream websites covering fifteen

¹Personal communication, Joel Reidenberg.

²<http://www.alexa.com>

Business	Computers	Games	Health
Home	News	Recreation	Shopping
Arts	Kids and Teens	Reference	Regional
Science	Society	Sports	

Table 1: Fifteen website categories provided by Alexa.com. We collect privacy policies from the top 100 websites in each.

of Alexa.com’s seventeen categories (Table 1).³

Finding a website’s policy is not trivial. Though many well-regulated commercial websites provide a “privacy” link on their homepages, not all do. We found university websites to be exceptionally unlikely to provide such a link. Even once the policy’s URL is identified, extracting the text presents the usual challenges associated with scraping documents from the web. Since every site is different in its placement of the document (e.g., buried deep within the website, distributed across several pages, or mingled together with Terms of Service) and format (e.g., HTML, PDF, etc.), and since we wish to preserve as much document structure as possible (e.g., section labels), full automation was not a viable solution.

We therefore crowdsourced the privacy policy document collection using Amazon Mechanical Turk. For each website, we created a HIT in which a worker was asked to copy and paste the following privacy policy-related information into text boxes: (i) privacy policy URL; (ii) last updated date (or effective date) of the current privacy policy; (iii) privacy policy full text; and (iv) the section subtitles in the top-most layer of the privacy policy. To identify the privacy policy URL, workers were encouraged to go to the website and search for the privacy link. Alternatively, they could form a search query using the website name and “privacy policy” (e.g., “Amazon.com privacy policy”) and search in the returned results for the most appropriate privacy policy URL. Given the privacy policy full text and the section subtitles, we partition the full privacy document into different sections, delimited by the section subtitles. A privacy policy is then converted into XML.

Each HIT was completed by three workers, paid \$0.05, for a total cost of \$380 (including Amazon’s surcharge).

³The “Adult” category was excluded; the “World” category was excluded since it contains mainly popular websites in different languages, and we opted to focus on policies in English in this first stage of research, though multilingual policy analysis presents interesting challenges for future work.

3 Approach

Given the corpus of privacy policies described in §2, we designed a model to efficiently infer an alignment of policy sections. While we expect that different kinds of websites will likely address different privacy issues, we believe that many policies will discuss roughly the same set of issues. Aligning the policies is a first step in a larger effort to (i) automatically analyze policies to make them less opaque to users and (ii) support legal experts who wish to characterize the state of privacy online and make recommendations (Costante et al., 2012; Ammar et al., 2012; Costante et al., 2013).

We are inspired by multiple sequence alignment methods in computational biology (Durbin et al., 1998) and by Barzilay and Lee (2004), who described a hidden Markov model (HMM) for document content where each state corresponds to a distinct topic and generates sentences relevant to that topic according to a language model. We estimate an HMM-like model on our corpus, exploiting similarity across privacy policies to the extent it is evident in the data. In our formulation, each hidden state corresponds to an issue or topic, characterized by a distribution over words and bigrams appearing in privacy policy sections addressing that issue. The transition distribution captures tendencies of privacy policy authors to organize these sections in similar orders, though with some variation.

The generative story for our model is as follows. Let \mathcal{S} denote the set of hidden states.

1. Choose a start state y_1 from \mathcal{S} according to the start-state distribution.
2. For $t = 1, 2, \dots$, until y_t is the stopping state:
 - (a) Sample the t th section of the document by drawing a bag of terms, \mathbf{o}_t , according to the emission multinomial distribution for state y_t . Note the difference from traditional HMMs, in which a *single* observation symbol is drawn at each time step. \mathbf{o}_t is generated by repeatedly sampling from a distribution over terms that includes all unigrams and bigrams except those that occur in fewer than 5% of the documents and in more than 98% of the documents. This filtering rule was designed to eliminate uninformative stopwords as well as company-specific terms (e.g., the name of the company).⁴

⁴The emission distributions are not a proper language

Category	Websites with privacy URL	Unique privacy policies	Unique privacy policies w/ date	Ave. sections per policy	Ave. tokens per policy
Arts	94	80	72	11.1 (\pm 3.8)	2894 (\pm 1815)
Business	100	95	75	10.1 (\pm 4.9)	2531 (\pm 1562)
Computers	100	78	62	10.7 (\pm 4.9)	2535 (\pm 1763)
Games	92	80	51	10.2 (\pm 4.9)	2662 (\pm 2267)
Health	92	86	57	10.0 (\pm 4.4)	2325 (\pm 1891)
Home	100	84	68	11.5 (\pm 3.8)	2493 (\pm 1405)
Kids and Teens	96	86	62	10.3 (\pm 4.5)	2683 (\pm 1979)
News	96	91	68	10.7 (\pm 3.9)	2588 (\pm 2493)
Recreation	98	97	67	11.9 (\pm 4.5)	2678 (\pm 1421)
Reference	84	86	55	9.9 (\pm 4.1)	2002 (\pm 1454)
Regional	98	91	72	11.2 (\pm 4.2)	2557 (\pm 1359)
Science	71	75	49	9.2 (\pm 4.1)	1705 (\pm 1136)
Shopping	100	99	84	12.0 (\pm 4.1)	2683 (\pm 1154)
Society	96	94	65	10.2 (\pm 4.6)	2505 (\pm 1587)
Sports	96	62	38	10.9 (\pm 4.0)	2222 (\pm 1241)
Average	94.2	85.6	63.0	10.7 (\pm 4.3)	2471 (\pm 1635)

Table 2: Statistics of each website category, including (i) the number of websites with an identified privacy policy link; (ii) number of unique privacy policies in each category (note that in rare cases, multiple unique privacy policies were identified for the same website, e.g., a website that contains links to both new and old versions of its privacy policy); (iii) number of websites with an identified privacy modification date; (iv) average number of sections per policy; (v) average number of tokens per policy.

- (b) Sample the next state, y_{t+1} , according to the transition distribution over \mathcal{S} .

This model can nearly be understood as a hidden *semi*-Markov model (Baum and Petrie, 1966), though we treat the section lengths as observable. Indeed, our model does not even generate these lengths, since doing so would force the states to “explain” the length of each section, not just its content. The likelihood function for the model is shown in Figure 1.

The parameters of the model are almost identical to those of a classic HMM (start state distribution, emission distributions, and transition distributions), except that emissions are characterized by multinomial rather than a categorical distributions. These are learned using Expectation-Maximization, with a forward-backward algorithm to calculate marginals (E-step) and smoothed maximum likelihood estimation for the M-step (Rabiner, 1989). After learning, the most probable assignment of a policy’s sections to states can be recovered using a variant of the Viterbi algorithm.

We consider three HMM variants. “Vanilla” allows all transitions. The other two posit an ordering on the states $\mathcal{S} = \{s_1, s_2, \dots, s_K\}$, and restrict the set of transitions that are possible, imposing bias on the learner. “All Forward” only allows

s_k to transition to $\{s_k, s_{k+1}, \dots, s_K\}$. “Strict Forward” only allows s_k to transition to s_k or s_{k+1} .

4 Evaluation

Developing a gold-standard alignment of privacy policies would either require an interface that allows each annotator to interact with the entire corpus of previously aligned documents while reading the one she is annotating, or the definition (and likely iterative refinement) of a set of categories for manually labeling policy sections. These were too costly for us to consider, so we instead propose two generic methods to evaluate models for sequence alignment of a collection of documents with generally similar content. Though our model (particularly the restricted variants) treats the problem as one of *alignment*, our evaluations consider *groupings* of policy sections. In the sequel, a grouping on a set X is defined as a collection of subsets $X_i \subseteq X$; these may overlap (i.e., there might be $x \in X_i \cap X_j$) and need not be exhaustive (i.e., there might be $x \in X \setminus \bigcup_i X_i$).

4.1 Evaluation by Human QA

This study was carried out as part of a larger collaboration with legal scholars who study privacy. In that work, we have formulated a set of nine multiple choice questions about a single policy that ask about collection of contact, location, health, and financial information, sharing of each with

models (e.g., a bigram may be generated by as many as three draws from the emission distribution: once for each unigram it contains and once for the bigram).

$$P_{\pi, \eta, \gamma} (\langle y_t, \mathbf{o}_t \rangle_{t=1}^n \mid \langle \ell_t \rangle_{t=1}^n) = \pi(y_1) \prod_{t=1}^n \left(\prod_{i=1}^{\ell_t} \eta(o_{t,i} \mid y_i) \right) \gamma(y_{t+1} \mid y_t)$$

Figure 1: The likelihood function for the alignment model (one privacy policy). y_t is the hidden state for the t th section, \mathbf{o}_t is the bag of unigram and bigram terms observed in that section, and ℓ_t is the size of the bag. Start-state, emission, and transition distributions are denoted respectively by π , η , and γ . y_{n+1} is the silent stopping state.

third parties, and deletion of data.⁵ The questions were inspired primarily by the substantive interest of these domain experts—not by this particular algorithmic study.

For thirty policies, we obtained answers from each of six domain experts who were not involved in designing the questions. For the purposes of this study, the experts’ answers are not important. In addition to answering each question for each policy, we also asked each expert to copy and paste the text of the policy that contains the answer. Experts were allowed to select as many sections for each question as they saw fit, since answering some questions may require synthesizing information from different sections.

For each of the nine questions, we take the union of all policy sections that contain text selected by any annotator as support for her answer. This results in nine groups of policy sections, which we call **answer-sets** denoted A_1, \dots, A_9 . Our method allows these to overlap (63% of the sections in any A_i occurred in more than one A_i), and they are not exhaustive (since many sections of the policies were not deemed to contain answers to any of the nine questions by any expert).

Together, these can be used as a gold standard grouping of policy sections, against which we can compare our system’s output. To do this, we define the set of section *pairs* that are grouped together in answer sets, $G = |\{\langle a, b \rangle \mid \exists A_i \ni a, b\}|$, and a similar set of pairs H from a model’s grouping. From these sets, we calculate estimates of precision ($|G \cap H|/|H|$) and recall ($|G \cap H|/|G|$).

One shortcoming of this approach, for which the second evaluation seeks to compensate, is that a very small, and likely biased, subset of the policy sections is considered.

4.2 Evaluation by Direct Judgment

We created a separate gold standard of judgments of pairs of privacy policy sections. The data selected for judgment was a sample of pairs stratified

⁵The questions are available in an online appendix at <http://usableprivacy.org/data>.

by a simple measure of text similarity. We derived unigram tfidf vectors for each section in each of 50 randomly sampled policies per category. We then binned *pairs* of sections by cosine similarity (into four bins bounded by 0.25, 0.5, and 0.75). We sampled 994 section pairs uniformly across the 15 categories’ four bins each.

Crowdsourcing was used to determine, for each pair, whether the two sections should be grouped together. A HIT consisted of a pair of policy sections and a multiple choice question, “After reading the two sections given below, would you say that they broadly discuss the same topic?” The possible answers were:

1. Yes, both the sections essentially convey the same message in a privacy policy.
2. Although, the sections do not convey the same message, the broadly discuss the same topic. (For ease of understanding, some examples of content on “the same topic” were included.)
3. No, the sections discuss two different topics.

The first two options were considered a “yes” for the majority voting and for defining a gold standard. Every section-pair was annotated by at least three annotators (as many as 15, increased until an absolute majority was reached). Turkers with an acceptance rate greater than 95% with an experience of at least 100 HITs were allowed and paid \$0.03 per annotation. The total cost including some initial trials was \$130. 535 out of the 994 pairs were annotated to be similar in topic. An example is shown in Figure 2.

As in §4.1, we calculate precision and recall on pairs. This does not penalize the model for grouping together a “no” pair; we chose it nonetheless because it is interpretable.

5 Experiment

In this section, we evaluate the three HMM variants described in §3, and two baselines, using the methods in §4. All of the methods require the specification of the number of groups or hidden states, which we fix to ten, the average number of sections per policy.

Section 5 of *classmates.com*:

[46 words] ... You may also be required to use a password to access certain pages on the Services where certain types of your personal information can be changed or deleted. ... [113 words]

Section 2 of *192.com*:

[50 words] ... This Policy sets out the means by which You can have Your Personal Information removed from the Service. 192.com is also committed to keeping Personal Information of users of the Service secure and only to use it for the purposes set out in this Policy and as agreed by You. ... [24 words]

Figure 2: Selections from sections that discuss the issue of “deletion of personal information” and were labeled as discussing the same issue by crowdworkers. Both naïve grouping and LDA put them in two different groups, but the Strict Forward variant of our model correctly groups them together.

	Precision		Recall		F_1	
	Mean	S.D.	Mean	S.D.	Mean	S.D.
Clust.	0.63	–	0.30	–	0.40	–
LDA	0.56	0.03	0.20	0.05	0.29	0.06
Vanilla	0.62	0.04	0.41	0.04	0.49	0.03
All F.	0.63	0.03	0.47	0.12	0.53	0.06
Strict F.	0.62	0.05	0.46	0.18	0.51	0.07
Clust.	0.62	–	0.23	–	0.34	–
LDA	0.57	0.03	0.18	0.01	0.28	0.02
Vanilla	0.57	0.01	0.30	0.03	0.39	0.02
All F.	0.58	0.02	0.32	0.06	0.41	0.04
Strict F.	0.58	0.03	0.32	0.14	0.40	0.08

Table 3: Evaluation by human QA (above) and direct judgment (below), aggregated across ten independent runs where appropriate (see text). Vanilla, All F(oward), and Strict F(oward) are three variants of our HMM.

Baselines. Our first baseline is a greedy divisive clustering algorithm⁶ to partition the policy sections into ten clusters. In this method, the desired K -way clustering solution is computed by performing a sequence of bisections. The implementation uses unigram features and cosine similarity. Our second baseline is latent Dirichlet allocation (LDA; Blei et al., 2003), with ten topics and online variational Bayes for inference (Hoffman et al., 2010).⁷ To more closely match our models, LDA is given access to the same unigram and bigram tokens.

Results. Table 3 shows the results. For LDA and the HMM variants (which use random initialization), we report mean and standard deviation across ten independent runs. All three variants of the HMM improve over the baselines on both tasks, in terms of F_1 . In the human QA evaluation, this is mostly due to recall improvements (i.e., more pairs of sections relevant to the same policy question were grouped together).

The three variants of the model performed similarly on average, though Strict Forward had very high variance. Its maximum performance across

⁶As implemented in CLUTO, <http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview>

⁷As implemented in gensim (Řehůřek and Sojka, 2010).

ten runs was very high (67% and 53% F_1 on the two tasks), suggesting the potential benefits of good initialization or model selection.

6 Conclusion

We considered the task of aligning sections of a collection of roughly similarly-structured legal documents, based on the issues they address. We introduced an unsupervised model for this task along with two new (and reusable) evaluations. Our experiments show the approach to be more effective than clustering and topic models. The corpus and evaluation data have been made available at <http://usableprivacy.org/data>. In future work, policy section alignments will be used in automated analysis to extract useful information for users and privacy scholars.

Acknowledgments

The authors gratefully acknowledge helpful comments from Lorrie Cranor, Joel Reidenberg, Florian Schaub, and several anonymous reviewers. This research was supported by NSF grant SaTC-1330596.

References

- Waleed Ammar, Shomir Wilson, Norman Sadeh, and Noah A. Smith. 2012. Automatic categorization of privacy policies: A pilot study. Technical Report CMU-LTI-12-019, Carnegie Mellon University.
- Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Proc. of HLT-NAACL*.
- Leonard E. Baum and Ted Petrie. 1966. Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37:1554–1563.
- David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

- Carolyn A. Brodie, Clare-Marie Karat, and John Karat. 2006. An empirical study of natural language parsing of privacy policy rules using the SPARCLE policy workbench. In *Proc. of the Symposium on Usable Privacy and Security*.
- Elisa Costante, Yuanhao Sun, Milan Petković, and Jerry den Hartog. 2012. A machine learning solution to assess privacy policy completeness. In *Proc. of the ACM Workshop on Privacy in the Electronic Society*.
- Elisa Costante, Jerry Hartog, and Milan Petkovi. 2013. What websites know about you. In Roberto Pietro, Javier Herranz, Ernesto Damiani, and Radu State, editors, *Data Privacy Management and Autonomous Spontaneous Security*, volume 7731 of *Lecture Notes in Computer Science*, pages 146–159. Springer Berlin Heidelberg.
- Richard Durbin, Sean R. Eddy, Anders Krogh, and Graeme Mitchison. 1998. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press.
- Federal Trade Commission. 2012. Protecting consumer privacy in an era of rapid change: Recommendations for businesses and policymakers.
- Robert Gellman. 2014. Fair information practices: a basic history (v. 2.11). Available at <http://www.bobgellman.com/rg-docs/rg-FIPShistory.pdf>.
- Matthew D Hoffman, David M Blei, and Francis R Bach. 2010. Online learning for latent Dirichlet allocation. In *NIPS*.
- Aleecia M. McDonald and Lorrie Faith Cranor. 2008. The cost of reading privacy policies. *IS: A Journal of Law and Policy for the Information Society*, 4(3).
- Lawrence Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.
- Radim Řehůřek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proc. of the LREC Workshop on New Challenges for NLP Frameworks*.
- Xusheng Xiao, Amit Paradkar, Suresh Thummalapenta, and Tao Xie. 2012. Automated extraction of security policies from natural-language software documents. In *Proc. of the ACM SIGSOFT International Symposium on the Foundations of Software Engineering*.
- Sebastian Zimmeck and Steven M. Bellovin. 2013. Machine learning for privacy policy.

Enriching Cold Start Personalized Language Model Using Social Network Information

Yu-Yang Huang[†], Rui Yan^{*}, Tsung-Ting Kuo[‡], Shou-De Lin^{†‡}

[†]Graduate Institute of Computer Science and Information Engineering,

National Taiwan University, Taipei, Taiwan

[‡]Graduate Institute of Network and Multimedia,

National Taiwan University, Taipei, Taiwan

^{*}Computer and Information Science Department,

University of Pennsylvania, Philadelphia, PA 19104, U.S.A.

{r02922050, d97944007, sdlin}@csie.ntu.edu.tw, ruiyan@seas.upenn.edu

Abstract

We introduce a generalized framework to enrich the personalized language models for cold start users. The cold start problem is solved with content written by friends on social network services. Our framework consists of a mixture language model, whose mixture weights are estimated with a factor graph. The factor graph is used to incorporate prior knowledge and heuristics to identify the most appropriate weights. The intrinsic and extrinsic experiments show significant improvement on cold start users.

1 Introduction

Personalized language models (PLM) on social network services are useful in many aspects (Xue et al., 2009; Wen et al., 2012; Clements, 2007). For instance, if the authorship of a document is in doubt, a PLM may be used as a generative model to identify it. In this sense, a PLM serves as a proxy of one’s writing style. Furthermore, PLMs can improve the quality of information retrieval and content-based recommendation systems, where documents or topics can be recommended based on the generative probabilities.

However, it is challenging to build a PLM for users who just entered the system, and whose content is thus insufficient to characterize them. These are called “cold start” users. Producing better recommendations is even more critical for cold start users to make them continue to use the system. Therefore, this paper focuses on how to overcome the cold start problem and obtain a better PLM for cold start users.

The content written by friends on a social network service, such as Facebook or Twitter, is exploited. It can be either a reply to an original post or posts by friends. Here the hypothesis is that friends, who usually share common interests,

tend to discuss similar topics and use similar words than non-friends. In other words, we believe that a cold start user’s language model can be *enriched* and better *personalized* by incorporating content written by friends.

Intuitively, a linear combination of document-level language models can be used to incorporate content written by friends. However, it should be noticed that some documents are more relevant than others, and should be weighted higher. To obtain better weights, some simple heuristics could be exploited. For example, we can measure the similarity or distance between a user language model and a document language model. In addition, documents that are shared frequently in a social network are usually considered to be more influential, and could contribute more to the language model. More complex heuristics can also be derived. For instance, if two documents are posted by the same person, their weights should be more similar. The main challenge lies in how such heuristics can be utilized in a systematic manner to infer the weights of each document-level language model.

In this paper, we exploit the information on social network services in two ways. First, we impose the *social dependency* assumption via a finite mixture model. We model the true, albeit unknown, personalized language model as a combination of a biased user language model and a set of relevant document language models. Due to the noise inevitably contained in social media content, instead of using all available documents, we argue that by properly specifying the set of relevant documents, a better personalized language model can be learnt. In other words, each user language model is enriched by a *personalized* collection of background documents.

Second, we propose a factor graph model (FGM) to incorporate prior knowledge (e.g. the heuristics described above) into our model. Each

mixture weight is represented by a random variable in the factor graph, and an efficient algorithm is proposed to optimize the model and infer the marginal distribution of these variables. Useful information about these variables is encoded by a set of potential functions.

The main contributions of this work are summarized below:

- To solve the cold start problem encountered when estimating PLMs, a generalized framework based on FGM is proposed. We incorporate social network information into user language models through the use of FGM. An iterative optimization procedure utilizing perplexity is presented to learn the parameters. To our knowledge, this is the first proposal to use FGM to enrich language models.
- Perplexity is selected as an intrinsic evaluation, and experiment on authorship attribution is used as an extrinsic evaluation. The results show that our model yields significant improvements for cold start users.

2 Methodology

2.1 Social-Driven Personalized Language Model

The language model of a collection of documents can be estimated by normalizing the counts of words in the entire collection (Zhai, 2008). To build a user language model, one naïve way is to first normalize word frequency $c(w, d)$ within each document, and then average over all the documents in a user’s document collection. The resulting unigram user language model is:

$$P_u(w) = \frac{1}{|\mathcal{D}_u|} \sum_{d \in \mathcal{D}_u} \frac{c(w, d)}{|d|} \quad (1)$$

$$= \frac{1}{|\mathcal{D}_u|} \sum_{d \in \mathcal{D}_u} P_d(w)$$

where $P_d(w)$ is the language model of a particular document, and \mathcal{D}_u is the user’s document collection. This formulation is basically an equal-weighted finite mixture model.

A simple yet effective way to smooth a language model is to linearly interpolate with a background language model (Chen and Goodman, 1996; Zhai and Lafferty, 2001). In the linear interpolation method, all background documents are treated equally. The entire document collection is added to the user language model $P_u(w)$ with the same interpolation coefficient.

Our main idea is to specify a set of relevant documents for the target user using information embedded in a social network, and enrich the

smoothing procedure with these documents. Let \mathcal{D}_{rel} denote the content from relevant persons (e.g. social neighbors) of u_i , our idea can be concisely expressed as:

$$P'_{u_1}(w) = \lambda_{u_1} P_{u_1}(w) + \sum_{d_i \in \mathcal{D}_{rel}} \lambda_{d_i} P_{d_i}(w) \quad (2)$$

where λ_{d_i} is the mixture weight of the language model of document d_i , and $\lambda_{u_1} + \sum \lambda_{d_i} = 1$. Documents posted by irrelevant users are not included as we believe the user language model can be better personalized by exploiting the social relationship in a more structured way. In our experiment, we choose the first degree neighbor documents as \mathcal{D}_{rel} .

Also note that we have made no assumption about how the “base” user language model $P_{u_1}(w)$ is built. In practice, it need not be models following maximum likelihood estimation, but any language model can be integrated into our framework to achieve a better refined model. Furthermore, any smoothing method can be applied to the language model without degrading the effectiveness.

2.2 Factor Graph Model (FGM)

Now we discuss how the mixture weights can be estimated. We introduce a *factor graph model* (FGM) to make use of the diverse information on a social network. Factor graph (Kschischang et al., 2006) is a bipartite graph consisting of a set of random variables and a set of factors which signifies the relationships among the variables. It is best suited in situations where the data is clearly of a relational nature (Wang et al., 2012). The joint distribution of the variables is factored according to the graph structure. Using FGM, one can incorporate the knowledge into the potential function for optimization and perform joint inference over documents. As shown in Figure 1, the variables included in the model are described as follows:

Candidate variables $y_i = \langle u, d_i \rangle$. The random variables in the top layer stand for the degrees of belief that a document d_i should be included in the PLM of the target user u .

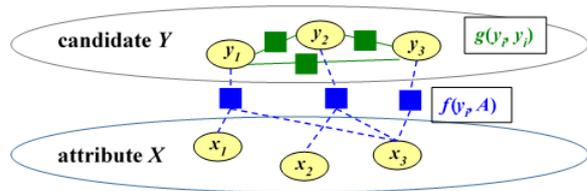


Figure 1: A two-layered factor graph (FGM) proposed to estimate the mixture weights.

Attribute variables x_i . Local information is stored as the random variables in the bottom layer. For example, x_i might represent the number of common friends between the author of a document d_i and our target user.

The potential functions in the FGM are:

Attribute-to-candidate function. This potential function captures the local dependencies of a candidate variable to the relevant attributes. Let the candidate variable y_i correspond to a document d_i , the attribute-to-candidate function of y_i is defined in a log-linear form:

$$f(y_i, A) = \frac{1}{Z_\alpha} \exp\{\alpha^T \mathbf{f}(y_i, A)\} \quad (3)$$

where A is the set of attributes of either the document d_i or target user u ; \mathbf{f} is a vector of feature functions which locally model the value of y_i with attributes in A ; Z_α is the local partition function and α is the weight vector to be learnt.

In our experiment, we define the vector of functions as $\mathbf{f} = \langle f_{sim}, f_{oov}, f_{pop}, f_{cmf}, f_{af} \rangle^T$ as:

- **Similarity function** f_{sim} . The similarity between language models of the target user and a document should play an important role. We use cosine similarity between two unigram models in our experiments.
- **Document quality function** f_{oov} . The out-of-vocabulary (OOV) ratio is used to measure the quality of a document. It is defined as

$$f_{oov} = 1 - \frac{|\{w: w \in d_i \cap w \notin V\}|}{|d_i|} \quad (4)$$

where V is the vocabulary set of the entire corpus, with stop words excluded.

- **Document popularity function** f_{pop} . This function is defined as the number of times d_i is shared to model the popularity of documents.
- **Common friend function** f_{cmf} . It is defined as the number of common friends between the target user u_i and the author of d_i .
- **Author friendship function** f_{af} . Assuming that documents posted by a user with more friends are more influential, this function is defined as the number of friends of d_i 's author.

Candidate-to-candidate function. This potential function defines the correlation of a candidate variable y_i with another candidate variable y_j in the factor graph. The function is defined as

$$g(y_i, y_j) = \frac{1}{Z_{ij, \beta}} \exp\{\beta^T \mathbf{g}(y_i, y_j)\} \quad (5)$$

where \mathbf{g} is a vector of feature functions indicating whether two variables are correlated. If we further denote the set of all related variables as

$G(y_i)$, then for any candidate variable y_i , we have the following brief expression:

$$g(y_i, G(y_i)) = \prod_{y_j \in G(y_i)} g(y_i, y_j) \quad (6)$$

For two candidate variables, let the corresponding document be d_i and d_j , respectively, we define the vector $\mathbf{g} = \langle g_{rel}, g_{cat} \rangle^T$ as:

- **User relationship function** g_{rel} . We assume that two candidate variables have higher dependency if they represent documents of the same author or the two authors are friends. The dependency should be even greater if two documents are similar. Let $a(d)$ denote the author of a document d and $\mathcal{N}[u]$ denote the closed neighborhood of a user u , we define

$$g_{rel} = \mathbb{I}\{a(d_j) \in \mathcal{N}[a(d_i)]\} \times sim(d_i, d_j) \quad (7)$$

- **Co-category function** g_{cat} . For any two candidate variables, it is intuitive that the two variables would have a higher correlation if d_i and d_j are of the same category. Let $c(d)$ denote the category of document d , we define

$$g_{cat} = \mathbb{I}\{c(d_i) = c(d_j)\} \times sim(d_i, d_j) \quad (8)$$

2.3 Model Inference and Optimization

Let Y and X be the set of all candidate variables and attribute variables, respectively. The joint distribution encoded by the FGM is given by multiplying all potential functions.

$$P(Y, X) = \prod_i f(y_i, A) g(y_i, G(y_i)) \quad (9)$$

The desired marginal distribution can be obtained by marginalizing all other variables. Since under most circumstances, however, the factor graph is densely connected, the exact inference is intractable and approximate inference is required. After obtaining the marginal probabilities, the mixture weights λ_{d_i} in Eq. 2 are estimated by normalizing the corresponding marginal probabilities $P(y_i)$ over all candidate variables, which can be written as

$$\lambda_{d_i} = (1 - \lambda_{u_1}) \frac{P(y_i)}{\sum_{j: d_j \in \mathcal{D}_{rel}} P(y_j)} \quad (10)$$

where the constraint $\lambda_{u_1} + \sum \lambda_{d_i} = 1$ leads to a valid probability distribution for our mixture model.

A factor graph is normally optimized by gradient-based methods. Unfortunately, since the ground truth values of the mixture weights are not available, we are prohibited from using supervised approaches. Here we propose a two-step iterative procedure to optimize our model. At

first, all the model parameters (i.e. α , β , λ_u) are randomly initialized. Then, we infer the marginal probabilities of candidate variables. Given these marginal probabilities, we can evaluate the perplexity of the user language model on a held-out dataset, and search for better parameters. This procedure is repeated until convergence. Also, notice that by using FGM, we reduce the number of parameters from $1 + |\mathcal{D}_{rel}|$ to $1 + |\alpha| + |\beta|$, lowering the risk of overfitting.

3 Experiments

3.1 Dataset and Experiment Setup

We perform experiments on the *Twitter* dataset collected by Galuba et al. (2010). Twitter data have been used to verify models with different purposes (Lin et al., 2011; Tan et al., 2011). To emphasize on the cold start scenario, we randomly selected 15 users with about 35 tweets and 70 friends as candidates for an authorship attribution task. Our experiment corpus consists of 4322 tweets. All words with less than 5 occurrences are removed. Stop words and URLs are also removed and all tweets are stemmed. We identify the 100 most frequent terms as categories. The size of the vocabulary set is 1377.

We randomly partitioned the tweets of each user into training, validation and testing sets. The reported result is the average of 10 random splits. In all experiments, we vary the size of training data from 1% to 15%, and hold out the same number of tweets from each user as validation and testing data. The statistics of our dataset, given 15% training data, are shown in Table 1.

Loopy belief propagation (LBP) is used to obtain the marginal probabilities of the variables (Murphy et al., 1999). Parameters are searched with the pattern search algorithm (Audet and Dennis, 2002). To not lose generality, we use the default configuration in all experiments.

# of	Max.	Min.	Avg.
Tweets	70	19	35.4
Friends	139	24	68.9
Variables	467	97	252.7
Edges	9216	231	3427.1

Table 1: Dataset statistics

3.2 Baseline Methods

We compare our framework with two baseline methods. The first (“*Cosine*”) is a straightforward implementation that sets all mixture weights λ_{d_i} to the cosine similarity between the probability mass vectors of the document and user unigram language models. The second (“*PS*”) uses the pattern search algorithm to perform constrained optimization over the mixture weights. As mentioned in section 2.3, the main difference between this method and ours (“*FGM*”) is that we reduce the search space of the parameters by FGM. Furthermore, social network information is exploited in our framework, while the PS method performs a direct search over mixture weights, discarding valuable knowledge.

Different from other smoothing methods that are usually mutually exclusive, any other smoothing methods can be easily merged into our framework. In Eq. 2, the *base language model* $P_{u_1}(w)$ can be already smoothed by any techniques before being plugged into our framework. Our framework then enriches the user language model with social network information. We select four popular smoothing methods to demonstrate such effect, namely additive smoothing, absolute smoothing (Ney et al., 1995), Jelinek-Mercer smoothing (Jelinek and Mercer, 1980) and Dirichlet smoothing (MacKay and Peto, 1994). The results of using only the base model (i.e. set $\lambda_{d_i} = 0$ in Eq. 2) are denoted as “*Base*” in the following tables.

Train %	Additive				Absolute			
	<u>Base</u>	<u>Cosine</u>	<u>PS</u>	<u>FGM</u>	<u>Base</u>	<u>Cosine</u>	<u>PS</u>	<u>FGM</u>
1%	900.4	712.6	725.5	537.5**	895.3	703.1	722.1	544.5**
5%	814.5	623.4	690.5	506.8**	782.4	607.9	678.4	510.2**
10%	757.7	566.6	684.8	481.2**	708.4	552.7	661.0	485.8**
15%	693.8	521.0	635.2	474.8**	647.4	504.3	622.3	474.1**
Train %	Jelinek-Mercer				Dirichlet			
	<u>Base</u>	<u>Cosine</u>	<u>PS</u>	<u>FGM</u>	<u>Base</u>	<u>Cosine</u>	<u>PS</u>	<u>FGM</u>
1%	637.8	571.4	643.1	541.0**	638.5	571.3	643.1	541.0**
5%	593.9	526.1	602.9	505.4**	595.0	526.6	616.5	507.2**
10%	559.2	494.1	573.8	483.6**	560.4	494.9	579.6	486.0**
15%	535.3	473.4	560.2	473.0	535.7	473.6	563.2	474.4

Table 2: Testing set perplexity. ** indicates that the best score among all methods is significantly better than the next highest score, by t-test at a significance level of 0.05.

3.3 Perplexity

As an intrinsic evaluation, we first compute the perplexity of unseen sentences under each user language model. The result is shown in Table 2.

Our method significantly outperforms all of the methods in almost all settings. We observe that the “PS” method takes a long time to converge and is prone to overfitting, likely because it has to search about a few hundred parameters on average. As expected, the advantage of our model is more apparent when the data is sparse.

3.4 Authorship Attribution (AA)

The authorship attribution (AA) task is chosen as the extrinsic evaluation metric. Here the goal is not about comparing with the state-of-the-art approaches in AA, but showing that LM-based approaches can benefit from our framework.

To apply PLM on this task, a naïve Bayes classifier is implemented (Peng et al., 2004). The most probable author of a document d is the one whose PLM yields the highest probability, and is determined by $u^* = \operatorname{argmax}_u \{\prod_{w \in d} P_u(w)\}$.

The result is shown in Table 3. Our model improves personalization and outperforms the baselines under cold start settings. When data is sparse, the “PS” method tends to overfit the noise, while the “Cosine” method contains too few information and is severely biased. Our method strikes a balance between model complexity and the amount of information included, and hence performs better than the others.

4 Related Work

Personalization has long been studied in various textual related tasks. Personalized search is established by modeling user behavior when using search engines (Shen et al., 2005; Xue et al., 2009). Query language model could be also expanded based on personalized user modeling

(Chirita et al., 2007). Personalization has also been modeled in many NLP tasks such as summarization (Yan et al., 2011) and recommendation (Yan et al., 2012). Different from our purpose, these models do not aim at exploiting social media content to enrich a language model. Wen et al. (2012) combines user-level language models from a social network, but instead of focusing on the cold start problem, they try to improve the speech recognition performance using a mass amount of texts on social network. On the other hand, our work explicitly models the more sophisticated document-level relationships using a probabilistic graphical model.

5 Conclusion

The advantage of our model is threefold. First, prior knowledge and heuristics about the social network can be adapted in a structured way through the use of FGM. Second, by exploiting a well-studied graphical model, mature inference techniques, such as LBP, can be applied in the optimization procedure, making it much more effective and efficient. Finally, different from most smoothing methods that are mutually exclusive, any other smoothing method can be incorporated into our framework to be further enhanced. Using only 1% of the training corpus, our model can improve the perplexity of base models by as much as 40% and the accuracy of authorship attribution by at most 15%.

6 Acknowledgement

This work was sponsored by AOARD grant number No. FA2386-13-1-4045 and National Science Council, National Taiwan University and Intel Corporation under Grants NSC102-2911-I-002-001 and NTU103R7501 and grant 102-2923-E-002-007-MY2, 102-2221-E-002-170, 101-2628-E-002-028-MY2.

Train %	Additive				Absolute			
	Base	Cosine	PS	FGM	Base	Cosine	PS	FGM
1%	54.67	58.27	61.07	63.74	49.47	57.60	58.27	64.27**
5%	61.47	63.20	62.67	68.40**	59.60	62.40	61.33	66.53**
10%	61.47	65.73	66.27	69.20**	61.47	65.20	64.67	71.87**
15%	64.27	67.07	62.13	70.40**	64.67	68.27	63.33	71.60**
Train %	Jelinek-Mercer				Dirichlet			
	Base	Cosine	PS	FGM	Base	Cosine	PS	FGM
1%	54.00	60.93	62.00	64.80**	52.80	60.40	61.87	64.67**
5%	62.67	65.47	64.00	68.00	60.80	65.33	62.40	66.93
10%	63.87	68.00	67.87	68.53	62.53	67.87	66.40	68.53
15%	65.87	70.40	64.14	69.87	65.47	70.27	64.53	68.40

Table 3: Accuracy (%) of authorship attribution. ** indicates that the best score among all methods is significantly better than the next highest score, by t-test at a significance level of 0.05.

Reference

- Charles Audet and J. E. Dennis, Jr. 2002. Analysis of generalized pattern searches. *SIAM J. on Optimization*, 13(3):889–903, August.
- Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, ACL '96, pages 310–318, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Paul Alexandru Chirita, Claudiu S. Firan, and Wolfgang Nejdl. 2007. Personalized query expansion for the web. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 7–14, New York, NY, USA. ACM.
- Maarten Clements. 2007. Personalization of social media. In *Proceedings of the 1st BCS IRSG Conference on Future Directions in Information Access*, FDIA'07, pages 14–14, Swinton, UK, UK. British Computer Society.
- Wojciech Galuba, Karl Aberer, Dipanjan Chakraborty, Zoran Despotovic, and Wolfgang Kellerer. 2010. Outtweeting the twitterers - predicting information cascades in microblogs. In *Proceedings of the 3rd Conference on Online Social Networks*, WOSN'10, pages 3–3, Berkeley, CA, USA. USENIX Association.
- Frederick Jelinek and Robert L. Mercer. 1980. Interpolated estimation of markov source parameters from sparse data. In *In Proceedings of the Workshop on Pattern Recognition in Practice*, pages 381–397, Amsterdam, The Netherlands: North-Holland, May.
- F. R. Kschischang, B. J. Frey, and H. A. Loeliger. 2006. Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theor.*, 47(2):498–519, September.
- Jimmy Lin, Rion Snow, and William Morgan. 2011. Smoothing techniques for adaptive online language models: Topic tracking in tweet streams. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 422–429, New York, NY, USA. ACM.
- David J.C. MacKay and Linda C. Bauman Peto. 1994. A hierarchical dirichlet language model. *Natural Language Engineering*, 1:1–19.
- Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. 1999. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, UAI'99, pages 467–475, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Hermann Ney, Ute Essen, and Reinhard Kneser. 1995. On the estimation of 'small' probabilities by leaving-one-out. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(12):1202–1212, December.
- Fuchun Peng, Dale Schuurmans, and Shaojun Wang. 2004. Augmenting naive bayes classifiers with statistical language models. *Inf. Retr.*, 7(3-4):317–345, September.
- Xuehua Shen, Bin Tan, and ChengXiang Zhai. 2005. Implicit user modeling for personalized search. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 824–831, New York, NY, USA. ACM.
- Chenhao Tan, Lillian Lee, Jie Tang, Long Jiang, Ming Zhou, and Ping Li. 2011. User-level sentiment analysis incorporating social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 1397–1405, New York, NY, USA. ACM.
- Zhichun Wang, Juanzi Li, Zhigang Wang, and Jie Tang. 2012. Cross-lingual knowledge linking across wiki knowledge bases. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 459–468, New York, NY, USA. ACM.
- Tsung-Hsien Wen, Hung-Yi Lee, Tai-Yuan Chen, and Lin-Shan Lee. 2012. Personalized language modeling by crowd sourcing with social network data for voice access of cloud applications. In *Spoken Language Technology Workshop (SLT), 2012 IEEE*, pages 188–193.
- Gui-Rong Xue, Jie Han, Yong Yu, and Qiang Yang. 2009. User language model for collaborative personalized search. *ACM Trans. Inf. Syst.*, 27(2):11:1–11:28, March.
- Rui Yan, Jian-Yun Nie, and Xiaoming Li. 2011. Summarize what you are interested in: An optimization framework for interactive personalized summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1342–1351, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Rui Yan, Mirella Lapata, and Xiaoming Li. 2012. Tweet recommendation with graph co-ranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 516–525, Stroudsburg, PA, USA. Association for Computational Linguistics.
- ChengXiang Zhai. 2008. *Statistical Language Models for Information Retrieval*. Now Publishers Inc., Hanover, MA, USA.

Chengxiang Zhai and John Lafferty. 2001. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 334–342, New York, NY, USA. ACM.

Automatic Labelling of Topic Models Learned from Twitter by Summarisation

Amparo Elizabeth Cano Basave[†] Yulan He[‡] Ruifeng Xu[§]

[†] Knowledge Media Institute, Open University, UK

[‡] School of Engineering and Applied Science, Aston University, UK

[§] Key Laboratory of Network Oriented Intelligent Computation

Shenzhen Graduate School, Harbin Institute of Technology, China

amparo.cano@open.ac.uk, y.he@cantab.net, xuruifeng@hitsz.edu.cn

Abstract

Latent topics derived by topic models such as Latent Dirichlet Allocation (LDA) are the result of hidden thematic structures which provide further insights into the data. The automatic labelling of such topics derived from social media poses however new challenges since topics may characterise novel events happening in the real world. Existing automatic topic labelling approaches which depend on external knowledge sources become less applicable here since relevant articles/concepts of the extracted topics may not exist in external sources. In this paper we propose to address the problem of automatic labelling of latent topics learned from Twitter as a summarisation problem. We introduce a framework which apply summarisation algorithms to generate topic labels. These algorithms are independent of external sources and only rely on the identification of dominant terms in documents related to the latent topic. We compare the efficiency of existing state of the art summarisation algorithms. Our results suggest that summarisation algorithms generate better topic labels which capture event-related context compared to the top- n terms returned by LDA.

1 Introduction

Topic model based algorithms applied to social media data have become a mainstream technique in performing various tasks including sentiment analysis (He, 2012) and event detection (Zhao et al., 2012; Diao et al., 2012). However, one of the main challenges is the task of understanding the semantics of a topic. This task has been approached by investigating methodologies for identifying meaningful topics through semantic coher-

ence (Aletras and Stevenson, 2013; Mimno et al., 2011; Newman et al., 2010) and for characterising the semantic content of a topic through automatic labelling techniques (Hulpus et al., 2013; Lau et al., 2011; Mei et al., 2007). In this paper we focus on the latter.

Our research task of automatic labelling a topic consists on selecting a set of words that best describes the semantics of the terms involved in this topic. The most generic approach to automatic labelling has been to use as primitive labels the top- n words in a topic distribution learned by a topic model such as LDA (Griffiths and Steyvers, 2004; Blei et al., 2003). Such top words are usually ranked using the marginal probabilities $P(w_i|t_j)$ associated with each word w_i for a given topic t_j . This task can be illustrated by considering the following topic derived from social media related to Education:

school protest student fee choic motherlod tuition teacher anger polic

where the top 10 words ranked by $P(w_i|t_j)$ for this topic are listed. Therefore the task is to find the top- n terms which are more representative of the given topic. In this example, the topic certainly relates to a student protest as revealed by the top 3 terms which can be used as a good label for this topic.

However previous work has shown that top terms are not enough for interpreting the coherent meaning of a topic (Mei et al., 2007). More recent approaches have explored the use of external sources (e.g. Wikipedia, WordNet) for supporting the automatic labelling of topics by deriving candidate labels by means of lexical (Lau et al., 2011; Magatti et al., 2009; Mei et al., 2007) or graph-based (Hulpus et al., 2013) algorithms applied on these sources.

Mei et al. (2007) proposed an unsupervised probabilistic methodology to automatically assign a label to a topic model. Their proposed approach

was defined as an optimisation problem involving the minimisation of the KL divergence between a given topic and the candidate labels while maximising the mutual information between these two word distributions. Lau et al. (2010) proposed to label topics by selecting top- n terms to label the overall topic based on different ranking mechanisms including pointwise mutual information and conditional probabilities.

Methods relying on external sources for automatic labelling of topics include the work by Maggati et al. (2009) which derived candidate topic labels for topics induced by LDA using the hierarchy obtained from the Google Directory service and expanded through the use of the OpenOffice English Thesaurus. Lau et al. (2011) generated label candidates for a topic based on top-ranking topic terms and titles of Wikipedia articles. They then built a Support Vector Regression (SVR) model for ranking the label candidates. More recently, Hulpus et al. (2013) proposed to make use of a structured data source (DBpedia) and employed graph centrality measures to generate semantic concept labels which can characterise the content of a topic.

Most previous topic labelling approaches focus on topics derived from well formatted and static documents. However in contrast to this type of content, the labelling of topics derived from tweets presents different challenges. In nature micro-post content is sparse and present ill-formed words. Moreover, the use of Twitter as the “what’s-happening-right now” tool, introduces new event-dependent relations between words which might not have a counter part in existing knowledge sources (e.g. Wikipedia). Our original interest in labelling topics stems from work in topic model based event extraction from social media, in particular from tweets (Shen et al., 2013; Diao et al., 2012). As opposed to previous approaches, the research presented in this paper addresses the labelling of topics exposing event-related content that might not have a counter part on existing external sources. Based on the observation that a short summary of a collection of documents can serve as a label characterising the collection, we propose to generate topic label candidates based on the summarisation of a topic’s relevant documents. Our contributions are two-fold:

- We propose a novel approach for topics labelling that relies on term relevance of documents

relating to a topic; and

- We show that summarisation algorithms, which are independent of external sources, can be used with success to label topics, presenting a higher performance than the top- n terms baseline.

2 Methodology

We propose to approach the topic labelling problem as a multi-document summarisation task. The following describes our proposed framework to characterise documents relevant to a topic.

2.1 Preliminaries

Given a set of documents the problem to be solved by topic modelling is the posterior inference of the variables, which determine the hidden thematic structures that best explain an observed set of documents. Focusing on the Latent Dirichlet Allocation (LDA) model (Blei et al., 2003; Griffiths and Steyvers, 2004), let \mathcal{D} be a corpus of documents denoted as $\mathcal{D} = \{\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_D\}$; where each document consists of a sequence of N_d words denoted by $\mathbf{d} = (w_1, w_2, \dots, w_{N_d})$; and each word in a document is an item from a vocabulary index of V different terms denoted by $\{1, 2, \dots, V\}$. Given D documents containing K topics expressed over V unique words, LDA generative process is described as follows:

- For each topic $k \in \{1, \dots, K\}$ draw $\phi_k \sim \text{Dirichlet}(\beta)$,
 - For each document $d \in \{1..D\}$:
 - ★ draw $\theta_d \sim \text{Dirichlet}(\alpha)$;
 - ★ For each word $n \in \{1..N_d\}$ in document d :
 - draw a topic $z_{d,n} \sim \text{Multinomial}(\theta_d)$;
 - draw a word $w_{d,n} \sim \text{Multinomial}(\varphi_{z_{d,n}})$.
- where ϕ_k is the word distribution for topic k , and θ_d is the distribution of topics in document d . Topics are interpreted using the top N terms ranked based on the marginal probability $p(w_i|t_j)$.

2.2 Automatic Labelling of Topic Models

Given K topics over the document collection \mathcal{D} , the topic labelling task consists on discovering a sequence of words for each topic $k \in \mathcal{K}$. We propose to generate topic label candidates by summarising topic relevant documents. Such documents can be derived using both the observed data from the corpus \mathcal{D} and the inferred topic model variables. In particular, the prominent topic of a document d can be found by

$$k_d = \arg \max_{k \in \mathcal{K}} p(k|d) \quad (1)$$

Therefore given a topic k , a set of C documents related to this topic can be obtained via equation 1.

Given the set of documents C relevant to topic k , we proposed to generate a label of a desired length x from the summarisation of C .

2.3 Topic Labelling by Summarisation

We compare different summarisation algorithms based on their ability to provide a good label to a given topic. In particular we investigate the use of lexical features by comparing three different well-known multi-document summarisation algorithms against the top- n topic terms baseline. These algorithms include:

Sum Basic (SB) This is a frequency based summarisation algorithm (Nenkova and Vanderwende, 2005), which computes initial word probabilities for words in a text. It then weights each sentence in the text (in our case a micropost) by computing the average probability of the words in the sentence. In each iteration it picks the highest weighted document and from it the highest weighted word. It uses an update function which penalises words which have already been picked.

Hybrid TFIDF (TFIDF) It is similar to SB, however rather than computing the initial word probabilities based on word frequencies it weights terms based on TFIDF. In this case the document frequency is computed as the number of times a word appears in a micropost from the collection C . Following the same procedure as SB it returns the top x weighted terms.

Maximal Marginal Relevance (MMR) This is a relevance based ranking algorithm (Carbonell and Goldstein, 1998), which avoids redundancy in the documents used for generating a summary. It measures the degree of dissimilarity between the documents considered and previously selected ones already in the ranked list.

Text Rank (TR) This is a graph-based summariser method (Mihalcea and Tarau, 2004) where each word is a vertex. The relevance of a vertex (term) to the graph is computed based on global information recursively drawn from the whole graph. It uses the PageRank algorithm (Brin and Page, 1998) to recursively change the weight of the vertices. The final score of a word is therefore not only dependent on the terms immediately connected to it but also on how these terms con-

nect to others. To assign the weight of an edge between two terms, TextRank computes word co-occurrence in windows of N words (in our case $N = 10$). Once a final score is calculated for each vertex of the graph, TextRank sorts the terms in a reverse order and provided the top T vertices in the ranking. Each of these algorithms produces a label of a desired length x for a given topic k .

3 Experimental Setup

3.1 Dataset

Our Twitter Corpus (TW) was collected between November 2010 and January 2011. TW comprises over 1 million tweets. We used the OpenCalais' document categorisation service¹ to generate categorical sets. In particular, we considered four different categories which contain many real-world events, namely: War and Conflict (War), Disaster and Accident (DisAc), Education (Edu) and Law and Crime (LawCri). The final TW dataset after removing retweets and short microposts (less than 5 words after removing stopwords) contains 7000 tweets in each category.

We preprocessed TW by first removing: punctuation, numbers, non-alphabet characters, stop words, user mentions, and URL links. We then performed Porter stemming (Porter, 1980) in order to reduce the vocabulary size. Finally to address the issue of data sparseness in the TW dataset, we removed words with a frequency lower than 5.

3.2 Generating the Gold Standard

Evaluation of automatic topic labelling often relied on human assessment which requires heavy manual effort (Lau et al., 2011; Hulpus et al., 2013). However performing human evaluations of Social Media test sets comprising thousands of inputs become a difficult task. This is due to both the corpus size, the diversity of event-related topics and the limited availability of domain experts. To alleviate this issue here, we followed the distribution similarity approach, which has been widely applied in the automatic generation of gold standards (GSs) for summary evaluations (Donaway et al., 2000; Lin et al., 2006; Louis and Nenkova, 2009; Louis and Nenkova, 2013). This approach compares two corpora, one for which no GS labels exist, against a reference corpus for which a GS exists. In our case these corpora correspond to the TW and a Newswire dataset (NW). Since previous

¹OpenCalais service, <http://www.opencalais.com>

research has shown that headlines are good indicators of the main focus of a text, both in structure and content, and that they can act as a human produced abstract (Nenkova, 2005), we used headlines as the GS labels of NW.

The News Corpus (NW) was collected during the same period of time as the TW corpus. NW consists of a collection of news articles crawled from traditional news media (BBC, CNN, and New York Times) comprising over 77,000 articles which include supplemental metadata (e.g. headline, author, publishing date). We also used the OpenCalais’ document categorisation service to automatically label news articles and considered the same four topical categories, (War, DisAc, Edu and LawCri). The same preprocessing steps were performed on NW.

Therefore, following a similarity alignment approach we performed the steps outlined in Algorithm 1 for generating the GS topic labels of a topic in TW.

Algorithm 1 GS for Topic Labels

Input: LDA topics for TW, and the LDA topics for NW for category c .

Output: Gold standard topic label for each of the LDA topics for TW.

```

1: for each topic  $i \in \{1, 2, \dots, 100\}$  from TW do
2:   for each topic  $j \in \{1, 2, \dots, 100\}$  from NW do
3:     Compute the Cosine similarity between word distributions of topic  $t_i$  and topic  $t_j$ .
4:   end for
5:   Select topic  $j$  which has the highest similarity to  $i$  and whose similarity measure is greater than a threshold (in this case 0.7)
6: end for
7: for each of the extracted topic pairs  $(t_i - t_j)$  do
8:   Collect relevant news articles  $\mathcal{C}_{NW}^j$  of topic  $t_j$  from the NW set.
9:   Extract the headlines of news articles from  $\mathcal{C}_{NW}^j$  and select the top  $x$  most frequent words as the gold standard label for topic  $t_i$  in the TW set
10: end for

```

These steps can be outlined as follows: 1) We ran LDA on TW and NW separately for each category with the number of topics set to 100; 2) We then aligned the Twitter topics and Newswire topics by the similarity measurement of word distributions of these topics (Ercan and Cicekli, 2008; Haghighi and Vanderwende, 2009; Wang et al., 2009; Delort and Alfonseca, 2012); 3) Finally to generate the GS label for each aligned topic pair $(t_i - t_j)$, we extracted the headlines of the news articles relevant to t_j and selected the top x most frequent words (after stop word removal and stemming). The generated label was used as the gold

standard label for the corresponding Twitter topic t_i in the topic pair.

4 Experimental Results

We compared the results of the summarisation techniques with the top terms (TT) of a topic as our baseline. These TT set corresponds to the top x terms ranked based on the probability of the word given the topic ($p(w|k)$) from the topic model. We evaluated these summarisation approaches with the ROUGE-1 method (Lin, 2004), a widely used summarisation evaluation metric that correlates well with human evaluation (Liu and Liu, 2008). This method measures the overlap of words between the generated summary and a reference, in our case the GS generated from the NW dataset.

The evaluation was performed at $x = \{1, \dots, 10\}$. Figure 1 presents the ROUGE-1 performance of the summarisation approaches as the length x of the generated topic label increases. We can see in all four categories that the SB and TFIDF approaches provide a better summarisation coverage as the length of the topic label increases. In particular, in both the Education and Law & Crime categories, both SB and TFIDF outperforms TT and TR by a large margin. The obtained ROUGE-1 performance is within the same range of performance previously reported on Social Media summarisation (Inouye and Kalita, 2011; Nichols et al., 2012; Ren et al., 2013).

Table 1 presents average results for ROUGE-1 in the four categories. Particularly the SB and TFIDF summarisation techniques consistently outperform the TT baseline across all four categories. SB gives the best results in three categories except War.

	ROUGE-1				
	TT	SB	TFIDF	MMR	TR
War	0.162	0.184	0.192	0.154	0.141
DisAc	0.134	0.194	0.160	0.132	0.124
Edu	0.106	0.240	0.187	0.104	0.023
LawCri	0.035	0.159	0.149	0.034	0.115

Table 1: Average ROUGE-1 for topic labels at $x = \{1..10\}$, generated from the TW dataset.

The generated labels with summarisation at $x = 5$ are presented in Table 2, where GS represents the label generated from the Newswire headlines.

Different summarisation techniques reveal words which do not appear in the top terms but

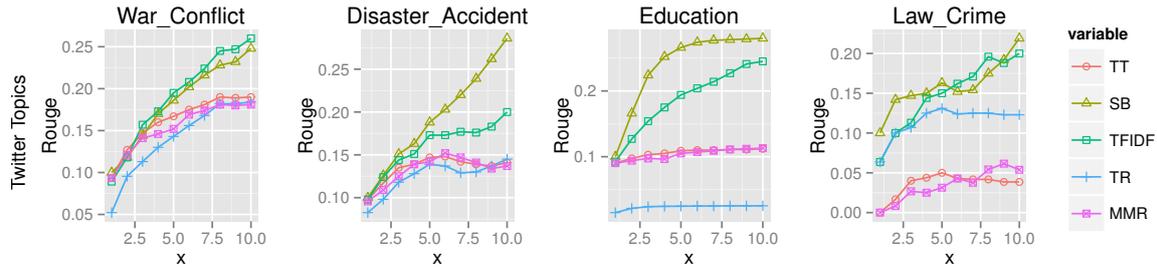


Figure 1: Performance in ROUGE for Twitter-derived topic labels, where x is the number of terms in the generated label

which are relevant to the information clustered by the topic. In this way, the labels generated for topics belonging to different categories generally extend the information provided by the top terms. For example in Table 2, the `DisAc` headline is characteristic of the New Zealand’s Pike River’s coal mine blast accident, which is an event occurred in November 2010.

Although the top 5 terms set from the LDA topic extracted from `TW` (listed under `TT`) does capture relevant information related to the event, it does not provide information regarding the blast. In this sense the topic label generated by `SB` more accurately describes this event.

We can also notice that the `GS` labels generated from Newswire media presented in Table 2 appear on their own, to be good labels for the `TW` topics. However as we described in the introduction we want to avoid relying on external sources for the derivation of topic labels.

This experiment shows that frequency based summarisation techniques outperform graph-based and relevance based summarisation techniques for generating topic labels that improve upon the top-terms baseline, without relying on external sources. This is an attractive property for automatically generating topic labels for tweets where their event-related content might not have a counter part on existing external sources.

5 Conclusions and Future Work

In this paper we proposed a novel alternative to topic labelling which do not rely on external data sources. To the best of our knowledge no existing work has been formally studied for automatic labelling through summarisation. This experiment shows that existing summarisation techniques can be exploited to provide a better label of a topic, extending in this way a topic’s information by pro-

	War	DisAc
GS	protest brief polic afghanistan attack world leader bomb obama pakistan	mine zealand rescu miner coal fire blast kill man dis- ast
<code>TT</code>	polic offic milit recent mosqu	mine coal pike river zealand
<code>SB</code>	terror war polic arrest offic	mine coal explos river pike
<code>TFIDF</code>	polic war arrest offic terror	mine coal pike safeti zealand
<code>MMR</code>	recent milit arrest attack target	trap zealand coal mine ex- plos
<code>TR</code>	war world peac terror hope	mine zealand plan fire fda
	Edu	LawCri
GS	school protest student fee choic motherlod tuition teacher anger polic	man charg murder arrest polic brief woman attack inquiri found
<code>TT</code>	student univers protest oc- cupi plan	man law child deal jail
<code>SB</code>	student univers school protest educ	man arrest law kill judg
<code>TFIDF</code>	student univers protest plan colleg	man arrest law judg kill
<code>MMR</code>	nation colleg protest stu- dent occupi	found kid wife student jail
<code>TR</code>	student tuition fee group hit	man law child deal jail

Table 2: Labelling examples for topics generated from the `TW` Dataset. `GS` represents the gold-standard generated from the relevant Newswire dataset. All terms are Porter stemmed as described in subsection 3.1

viding a richer context than top-terms. These results show that there is room to further improve upon existing summarisation techniques to cater for generating candidate labels.

Acknowledgments

This work was supported by the EPSRC grant EP/J020427/1, the EU-FP7 project SENSE4US (grant no. 611242), and the Shenzhen International Cooperation Research Funding (grant number GJHZ20120613110641217).

References

- Nikolaos Aletras and Mark Stevenson. 2013. Evaluating topic coherence using distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Long Papers*, pages 13–22, Potsdam, Germany, March. Association for Computational Linguistics.
- David Meir Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. In *J. Mach. Learn. Res.* 3, pages 993–1022.
- Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine*. 1. In *Computer networks and ISDN systems*, volume 30, pages 107–117.
- 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.
- Jean-Yves Delort and Enrique Alfonseca. 2012. Dual-sum: A topic-model based approach for update summarization. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 214–223, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Qiming Diao, Jing Jiang, Feida Zhu, and Ee-Peng Lim. 2012. Finding bursty topics from microblogs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 536–544, Jeju Island, Korea, July. Association for Computational Linguistics.
- Robert L. Donaway, Kevin W. Drummey, and Laura A. Mather. 2000. A comparison of rankings produced by summarization evaluation measures. In *Proceedings of the 2000 NAACL-ANLP Workshop on Automatic Summarization*, NAACL-ANLP-AutoSum '00, pages 69–78, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Gonenc Ercan and Ilyas Cicekli. 2008. Lexical cohesion based topic modeling for summarization. In *Proceedings of the 9th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing'08, pages 582–592, Berlin, Heidelberg. Springer-Verlag.
- Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *PNAS*, 101(suppl. 1):5228–5235.
- 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*, NAACL '09, pages 362–370, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Yulan He. 2012. Incorporating sentiment prior knowledge for weakly supervised sentiment analysis. *ACM Transactions on Asian Language Information Processing*, 11(2):4:1–4:19, June.
- Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. 2013. Unsupervised graph-based topic labelling using dbpedia. In *Proceedings of the sixth ACM international conference on Web search and data mining*, WSDM '13, pages 465–474, New York, NY, USA. ACM.
- David Inouye and Jugal K. Kalita. 2011. Comparing twitter summarization algorithms for multiple post summaries. In *SocialCom/PASSAT*, pages 298–306. IEEE.
- Jey Han Lau, David Newman, Karimi Sarvnaz, and Timothy Baldwin. 2010. Best Topic Word Selection for Topic Labelling. *CoLing*.
- Jey Han Lau, Karl Grieser, David Newman, and Timothy Baldwin. 2011. Automatic labelling of topic models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1536–1545, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Chin-Yew Lin, Guihong Cao, Jianfeng Gao, and Jian-Yun Nie. 2006. An information-theoretic approach to automatic evaluation of summaries. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 463–470, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Stan Szpakowicz Marie-Francine Moens, editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain, July. Association for Computational Linguistics.
- Feifan Liu and Yang Liu. 2008. Correlation between rouge and human evaluation of extractive meeting summaries. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 201–204, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Annie Louis and Ani Nenkova. 2009. Automatically evaluating content selection in summarization without human models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 306–314, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Annie Louis and Ani Nenkova. 2013. Automatically assessing machine summary content without a gold

- standard. *Computational Linguistics*, 39(2):267–300.
- Davide Magatti, Silvia Calegari, Davide Ciucci, and Fabio Stella. 2009. Automatic labeling of topics. In *Proceedings of the 2009 Ninth International Conference on Intelligent Systems Design and Applications*, ISDA '09, pages 1227–1232, Washington, DC, USA. IEEE Computer Society.
- Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 490–499, New York, NY, USA. ACM.
- Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Texts. In *Conference on Empirical Methods in Natural Language Processing*, EMNLP '04, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.
- David Mimno, Hanna M. Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 2011. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 262–272, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ani Nenkova and Lucy Vanderwende. 2005. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005-101*.
- Ani Nenkova. 2005. Automatic text summarization of newswire: Lessons learned from the document understanding conference. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 3*, AAAI'05, pages 1436–1441. AAAI Press.
- David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. 2010. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 100–108, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Jeffrey Nichols, Jalal Mahmud, and Clemens Drews. 2012. Summarizing sporting events using twitter. In *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces*, IUI '12, pages 189–198, New York, NY, USA. ACM.
- Martin Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.
- Zhaochun Ren, Shangsong Liang, Edgar Meij, and Maarten de Rijke. 2013. Personalized time-aware tweets summarization. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 513–522, New York, NY, USA. ACM.
- Chao Shen, Fei Liu, Fuliang Weng, and Tao Li. 2013. A participant-based approach for event summarization using twitter streams. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '13, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Dingding Wang, Shenghuo Zhu, Tao Li, and Yihong Gong. 2009. Multi-document summarization using sentence-based topic models. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 297–300, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Xin Zhao, Baihan Shu, Jing Jiang, Yang Song, Hongfei Yan, and Xiaoming Li. 2012. Identifying event-related bursts via social media activities. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1466–1477, Jeju Island, Korea, July. Association for Computational Linguistics.

Stochastic Contextual Edit Distance and Probabilistic FSTs

Ryan Cotterell and Nanyun Peng and Jason Eisner
Department of Computer Science, Johns Hopkins University
{ryan.cotterell, npeng1, jason}@cs.jhu.edu

Abstract

String similarity is most often measured by weighted or unweighted edit distance $d(x, y)$. Ristad and Yianilos (1998) defined *stochastic* edit distance—a probability distribution $p(y | x)$ whose parameters can be trained from data. We generalize this so that the probability of choosing each edit operation can depend on contextual features. We show how to construct and train a probabilistic finite-state transducer that computes our stochastic contextual edit distance. To illustrate the improvement from conditioning on context, we model typos found in social media text.

1 Introduction

Many problems in natural language processing can be viewed as stochastically mapping one string to another: e.g., transliteration, pronunciation modeling, phonology, morphology, spelling correction, and text normalization. Ristad and Yianilos (1998) describe how to train the parameters of a stochastic editing process that moves through the input string x from left to right, transforming it into the output string y . In this paper we generalize this process so that the edit probabilities are conditioned on input and output context.

We further show how to model the conditional distribution $p(y | x)$ as a probabilistic finite-state transducer (PFST), which can be easily combined with other transducers or grammars for particular applications. We contrast our *probabilistic* transducers with the more general framework of *weighted* finite-state transducers (WFST), explaining why our restriction provides computational advantages when reasoning about unknown strings.

Constructing the finite-state transducer is tricky, so we give the explicit construction for use by others. We describe how to train its parameters when the contextual edit probabilities are given by a log-linear model. We provide a library for training both PFSTs and WFSTs that works with OpenFST (Allauzen et al., 2007), and we illustrate its use with simple experiments on typos, which demonstrate the benefit of context.

2 Stochastic Contextual Edit Distance

Our goal is to define a family of probability distributions $p_\theta(y | x)$, where $x \in \Sigma_x^*$ and $y \in \Sigma_y^*$ are input and output strings over finite alphabets Σ_x and Σ_y , and θ is a parameter vector.

Let x_i denote the i^{th} character of x . If $i < 1$ or $i > |x|$, then x_i is the distinguished symbol BOS or EOS (“beginning/end of string”). Let $x_{i:j}$ denote the $(j - i)$ -character substring $x_{i+1}x_{i+2} \cdots x_j$.

Consider a stochastic edit process that reads input string x while writing output string y . Having read the prefix $x_{0:i}$ and written the prefix $y_{0:j}$, the process must stochastically choose one of the following $2|\Sigma_y| + 1$ edit operations:

- DELETE: Read x_{i+1} but write nothing.
- INSERT(t) for some $t \in \Sigma_y$: Write $y_{j+1} = t$ without reading anything.
- SUBST(t) for some $t \in \Sigma_y$: Read x_{i+1} and write $y_{j+1} = t$. Note that the traditional COPY operation is obtained as SUBST(x_{i+1}).

In the special case where $x_{i+1} = \text{EOS}$, the choices are instead INSERT(t) and HALT (where the latter may be viewed as copying the EOS symbol).

The probability of each edit operation depends on θ and is conditioned on the left input context $C_1 = x_{(i-N_1):i}$, the right input context $C_2 = x_{i:(i+N_2)}$, and the left output context $C_3 = y_{(j-N_3):j}$, where the constants $N_1, N_2, N_3 \geq 0$ specify the model’s context window sizes.¹ Note that the probability cannot be conditioned on right output context because those characters have not yet been chosen. Ordinary stochastic edit distance (Ristad and Yianilos, 1998) is simply the case $(N_1, N_2, N_3) = (0, 1, 0)$, while Bouchard-Côté et al. (2007) used roughly $(1, 2, 0)$.

Now $p_\theta(y | x)$ is the probability that this process will write y as it reads a given x . This is the total probability (given x) of *all* latent edit operation sequences that write y . In general there are exponentially many such sequences, each implying a different alignment of y to x .

¹If $N_2 = 0$, so that we do not condition on x_{i+1} , we must still condition on whether $x_{i+1} = \text{EOS}$ (a single bit). We gloss over special handling for $N_2 = 0$; but it is in our code.

This model is reminiscent of conditional models in MT that perform stepwise generation of one string or structure from another—e.g., string alignment models with contextual features (Cherry and Lin, 2003; Liu et al., 2005; Dyer et al., 2013), or tree transducers (Knight and Graehl, 2005).

3 Probabilistic FSTs

We will construct a **probabilistic finite-state transducer (PFST)** that compactly models $p_\theta(y | x)$ for all (x, y) pairs.² Then various computations with this distribution can be reduced to standard finite-state computations that efficiently employ dynamic programming over the structure of the PFST, and the PFST can be easily combined with other finite-state distributions and functions (Mohri, 1997; Eisner, 2001).

A PFST is a two-tape generalization of the well-known nondeterministic finite-state acceptor. It is a finite directed multigraph where each arc is labeled with an input in $\Sigma_x \cup \{\epsilon\}$, an output in $\Sigma_y \cup \{\epsilon\}$, and a probability in $[0, 1]$. (ϵ is the empty string.) Each state (i.e., vertex) has a halt probability in $[0, 1]$, and there is a single initial state q_I . Each path from q_I to a final state q_F has

- an input string x , given by the concatenation of its arcs’ input labels;
- an output string y , given similarly;
- a probability, given by the product of its arcs’ probabilities and the halt probability of q_F .

We define $p(y | x)$ as the total probability of all paths having input x and output y . In our application, a PFST path corresponds to an edit sequence that reads x and writes y . The path’s probability is the probability of that edit sequence given x .

We must take care to ensure that for any $x \in \Sigma_x^*$, the total probability of all paths accepting x is 1, so that $p_\theta(y | x)$ is truly a conditional probability distribution. This is guaranteed by the following sufficient conditions (we omit the proof for space), which do not seem to appear in previous literature:

- For each state q and each symbol $b \in \Sigma_x$, the arcs from q with input label b or ϵ must have total probability of 1. (These are the available choices if the next input character is x .)

²Several authors have given recipes for finite-state transducers that perform a *single* contextual edit operation (Kaplan and Kay, 1994; Mohri and Sproat, 1996; Gerdemann and van Noord, 1999). Such “rewrite rules” can be individually more expressive than our simple edit operations of section 2; but it is unclear how to train a cascade of them to model $p(y | x)$.

- For each state q , the halt action and the arcs from q with input label ϵ must have total probability of 1. (These are the available choices if there is no next input character.)
- Every state q must be co-accessible, i.e., there must be a path of probability > 0 from q to some q_F . (Otherwise, the PFST could lose some probability mass to infinite paths. The canonical case of this involves an loop $q \rightarrow q$ with input label ϵ and probability 1.)

We take the first two conditions to be part of the *definition* of a PFST. The final condition requires our PFST to be “tight” in the same sense as a PCFG (Chi and Geman, 1998), although the tightness conditions for a PCFG are more complex. In section 7, we discuss the costs and benefits of PFSTs relative to other options.

4 The Contextual Edit PFST

We now define a PFST topology that concisely captures the contextual edit process of section 2. We are given the alphabets Σ_x, Σ_y and the context window sizes $N_1, N_2, N_3 \geq 0$.

For each possible context triple $C = (C_1, C_2, C_3)$ as defined in section 2, we construct an **edit state** q_C whose outgoing arcs correspond to the possible edit operations in that context.

One might expect that the $\text{SUBST}(t)$ edit operation that reads $s = x_{i+1}$ and writes $t = y_{j+1}$ would correspond to an arc with s, t as its input and output labels. However, we give a more efficient design where in the course of reaching q_C , the PFST has *already* read s and indeed the entire right input context $C_2 = x_{i:(i+N_2)}$. So our PFST’s input and output actions are “out of sync”: its read head is N_2 characters ahead of its write head. When the edit process of section 2 has read $x_{0:i}$ and written $y_{0:j}$, our PFST implementation will actually have read $x_{0:(i+N_2)}$ and written $y_{0:j}$.

This design eliminates the need for nondeterministic guessing (of the right context $x_{i:(i+N_2)}$) to determine the edit probability. The PFST’s state is fully determined by the characters that it has read and written so far. This makes left-to-right composition in section 5 efficient.

A fragment of our construction is illustrated in Figure 1. An edit state q_C has the following outgoing **edit arcs**, each of which corresponds to an edit operation that replaces some $s \in \Sigma_x \cup \{\epsilon\}$ with some $t \in \Sigma_y \cup \{\epsilon\}$:

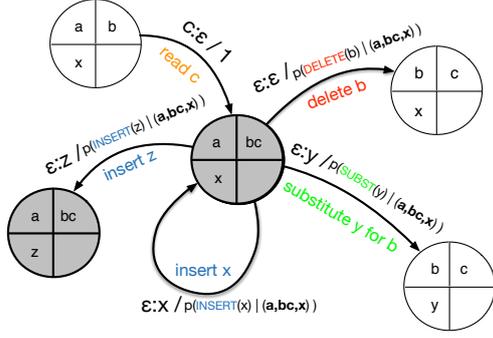


Figure 1: A fragment of a PFST with $N_1 = 1, N_2 = 2, N_3 = 1$. Edit states are shaded. A state q_C is drawn with left and right input contexts C_1, C_2 in the left and right upper quadrants, and left output context C_3 in the left lower quadrant. Each arc is labeled with input:output / probability.

- A single arc with probability $p(\text{DELETE} | C)$ (here $s = (C_2)_1, t = \epsilon$)
- For each $t \in \Sigma_y$, an arc with probability $p(\text{INSERT}(t) | C)$ (here $s = \epsilon$)
- For each $t \in \Sigma_y$, an arc with probability $p(\text{SUBST}(t) | C)$ (here $s = (C_2)_1$)

Each edit arc is labeled with input ϵ (because s has *already* been read) and output t . The arc leads from q_C to $q_{C'}$, a state that moves s and t into the left contexts: $C'_1 = \text{suffix}(C_1 s, N_1)$, $C'_2 = \text{suffix}(C_2, N_2 - |s|)$, $C'_3 = \text{suffix}(C_3 t, N_3)$.

Section 2 mentions that the end of x requires special handling. An edit state q_C whose $C_2 = \text{EOS}^{N_2}$ only has outgoing $\text{INSERT}(t)$ arcs, and has a halt probability of $p(\text{HALT} | C)$. The halt probability at all other states is 0.

We must also build some **non-edit states** of the form q_C where $|C_2| < N_2$. Such a state does not have the full N_2 characters of lookahead that are needed to determine the conditional probability of an edit. Its outgoing arcs deterministically read a new character into the right input context. For each $s \in \Sigma_x$, we have an arc of probability 1 from q_C to $q_{C'}$ where $C' = (C_1, C_2 s, C_3)$, labeled with input s and output ϵ . Following such arcs from q_C will reach an edit state after $N_2 - |C_2|$ steps.

The initial state q_I with $I = (\text{BOS}^{N_1}, \epsilon, \text{BOS}^{N_3})$ is a non-edit state. Other non-edit states are constructed only when they are reachable from another state. In particular, a DELETE or SUBST arc always transitions to a non-edit state, since it consumes one of the lookahead characters.

5 Computational Complexity

We summarize some useful facts without proof. For fixed alphabets Σ_x and Σ_y , our final

PFST, T , has $O(|\Sigma_x|^{N_1+N_2} |\Sigma_y|^{N_3})$ states and $O(|\Sigma_x|^{N_1+N_2} |\Sigma_y|^{N_3+1})$ arcs. Composing this T with deterministic FSAs takes time linear in the size of the *result*, using a left-to-right, on-the-fly implementation of the composition operator \circ .

Given strings x and y , we can compute $p_\theta(y | x)$ as the total probability of all paths in $x \circ T \circ y$. This acyclic weighted FST has $O(|x| \cdot |y|)$ states and arcs. It takes only $O(|x| \cdot |y|)$ time to construct it and sum up its paths by dynamic programming, just as in other edit distance algorithms.

Given only x , taking the output language of $x \circ T$ yields the full distribution $p_\theta(y | x)$ as a cyclic PFSA with $O(|x| \cdot \Sigma_y^{N_3})$ states and $O(|x| \cdot \Sigma_y^{N_3+1})$ arcs. Finding its most probable path (i.e., most probable aligned y) takes time $O(|\text{arcs}| \log |\text{states}|)$, while computing every arc's expected number of traversals under $p(y | x)$ takes time $O(|\text{arcs}| \cdot |\text{states}|)$.³

$p_\theta(y | x)$ may be used as a noisy channel model. Given a language model $p(x)$ represented as a PFSA X , $X \circ T$ gives $p(x, y)$ for all x, y . In the case of an n -gram language model with $n \leq N_1 + N_2$, this composition is efficient: it merely reweights the arcs of T . We use Bayes' Theorem to reconstruct x from observed y : $X \circ T \circ y$ gives $p(x, y)$ (proportional to $p(x | y)$) for each x . This weighted FSA has $O(\Sigma_x^{N_1+N_2} \cdot |y|)$ states and arcs.

6 Parameterization and Training

While the parameters θ could be trained via various objective functions, it is particularly efficient to compute the gradient of *conditional log-likelihood*, $\sum_k \log p_\theta(y_k | x_k)$, given a sample of pairs (x_k, y_k) . This is a non-convex objective function because of the latent x -to- y alignments: we do not observe *which* path transduced x_k to y_k . Recall from section 5 that these possible paths are represented by the small weighted FSA $x_k \circ T \circ y_k$.

Now, a path's probability is defined by multiplying the contextual probabilities of edit operations e . As suggested by Berg-Kirkpatrick et al. (2010), we model these steps using a conditional log-linear model, $p_\theta(e | C) \stackrel{\text{def}}{=} \frac{1}{Z_C} \exp(\theta \cdot \vec{f}(C, e))$.

³Speedups: In both runtimes, a factor of $|x|$ can be eliminated from $|\text{states}|$ by first decomposing $x \circ T$ into its $O(|x|)$ strongly connected components. And the $|\text{states}|$ factor in the second runtime is unnecessary in practice, as just the first few iterations of conjugate gradient are enough to achieve good approximate convergence when solving the sparse linear system that defines the forward probabilities in the cyclic PFSA.

To increase $\log p_\theta(y_k | x_k)$, we must raise the probability of the edits e that were used to transduce x_k to y_k , relative to competing edits from the same contexts C . This means raising $\theta \cdot f(C, e)$ and/or lowering Z_C . Thus, $\log p_\theta(y_k | x_k)$ depends only on the probabilities of edit arcs in T that appear in $x_k \circ T \circ y_k$, and the competing edit arcs from the same edit states q_C .

The gradient $\nabla_\theta \log p_\theta(y_k | x_k)$ takes the form

$$\sum_{C,e} c(C, e) \left[\vec{f}(C, e) - \sum_{e'} p_\theta(e' | C) \vec{f}(C, e') \right]$$

where $c(C, e)$ is the *expected* number of times that e was chosen in context C given (x_k, y_k) . (That can be found by the forward-backward algorithm on $x_k \circ T \circ y_k$.) So the gradient adds up the differences between observed and expected feature vectors at contexts C , where contexts are weighted by how many times they were likely encountered.

In practice, it is efficient to hold the counts $c(C, e)$ constant over several gradient steps, since this amortizes the work of computing them. This can be viewed as a generalized EM algorithm that imputes the hidden paths (giving c) at the ‘‘E’’ step and improves their probability at the ‘‘M’’ step.

Algorithm 1 provides the training pseudocode.

Algorithm 1 Training a PFST T_θ by EM.

```

1: while not converged do
2:   reset all counts to 0 ▷ begin the ‘‘E step’’
3:   for  $k \leftarrow 1$  to  $K$  do ▷ loop over training data
4:      $M = x_k \circ T_\theta \circ y_k$  ▷ small acyclic WFST
5:      $\vec{\alpha} = \text{FORWARD-ALGORITHM}(M)$ 
6:      $\vec{\beta} = \text{BACKWARD-ALGORITHM}(M)$ 
7:     for arc  $A \in M$ , from state  $q \rightarrow q'$  do
8:       if  $A$  was derived from an arc in  $T_\theta$ 
9:         representing edit  $e$ , from edit state  $q_C$ , then
10:         $c(C, e) += \alpha_q \cdot \text{prob}(A) \cdot \beta_{q'}/\beta_{q_1}$ 
11:    $\theta \leftarrow \text{L-BFGS}(\theta, \text{EVAL}, \text{max\_iters}=5)$  ▷ the ‘‘M step’’
12:   function  $\text{EVAL}(\theta)$  ▷ objective function & its gradient
13:      $F \leftarrow 0$ ;  $\nabla F \leftarrow 0$ 
14:     for context  $C$  such that  $(\exists e)c(C, e) > 0$  do
15:        $\text{count} \leftarrow 0$ ;  $\text{expected} \leftarrow 0$ ;  $Z_C \leftarrow 0$ 
16:       for possible edits  $e$  in context  $C$  do
17:          $F += c(C, e) \cdot (\theta \cdot \vec{f}(C, e))$ 
18:          $\nabla F += c(C, e) \cdot \vec{f}(C, e)$ 
19:          $\text{count} += c(C, e)$ 
20:          $\text{expected} += \exp(\theta \cdot \vec{f}(C, e)) \cdot \vec{f}(C, e)$ 
21:          $Z_C += \exp(\theta \cdot \vec{f}(C, e))$ 
22:        $F -= \text{count} \cdot \log Z_C$ ;  $\nabla F -= \text{count} \cdot \text{expected}/Z_C$ 
23:   return  $(F, \nabla F)$ 

```

7 PFSTs versus WFSTs

Our PFST model of $p(y | x)$ enforces a normalized probability distribution at each state. Drop-

ping this requirement gives a **weighted FST (WFST)**, whose path weights $w(x, y)$ can be globally normalized (divided by a constant Z_x) to obtain probabilities $p(y | x)$. WFST models of contextual edits were studied by Dreyer et al. (2008).

PFSTs and WFSTs are respectively related to MEMMs (McCallum et al., 2000) and CRFs (Lafferty et al., 2001). They gain added power from hidden states and ϵ transitions (although to permit a *finite*-state encoding, they condition on x in a more restricted way than MEMMs and CRFs).

WFSTs are likely to beat PFSTs as linguistic models,⁴ just as CRFs beat MEMMs (Klein and Manning, 2002). A WFST’s advantage is that the probability of an edit can be *indirectly* affected by the weights of other edits at a distance. Also, one could construct WFSTs where an edit’s weight *directly* considers local right output context C_4 .

So why are we interested in PFSTs? Because they do not require computing a separate normalizing constant Z_x for every x . This makes it computationally tractable to use them in settings where x is uncertain because it is unobserved, partially observed (e.g., lacks syllable boundaries), or noisily observed. E.g., at the end of section 5, X represented an uncertain x . So unlike WFSTs, PFSTs are usable as the conditional distributions in noisy channel models, channel cascades, and Bayesian networks. In future we plan to measure their modeling disadvantage and attempt to mitigate it.

PFSTs are also more efficient to train under conditional likelihood. It is faster to compute the gradient (and fewer steps seem to be required in practice), since we only have to raise the probabilities of arcs in $x_k \circ T \circ y_k$ relative to competing arcs in $x_k \circ T$. We visit at most $|x_k| \cdot |y_k| \cdot |\Sigma_y|$ arcs. By contrast, training a WFST must raise the probability of the *paths* in $x_k \circ T \circ y_k$ relative to the *infinitely many* competing paths in $x_k \circ T$. This requires summing around cycles in $x_k \circ T$, and requires visiting all of its $|x_k| \cdot |\Sigma_y|^{N_3+1}$ arcs.

8 Experiments

To demonstrate the utility of contextual edit transducers, we examine spelling errors in social media data. Models of spelling errors are useful in a variety of settings including spelling correction itself and phylogenetic models of string variation

⁴WFSTs can also use a simpler topology (Dreyer et al., 2008) while retaining determinism, since edits can be scored ‘‘in retrospect’’ after they have passed into the left context.

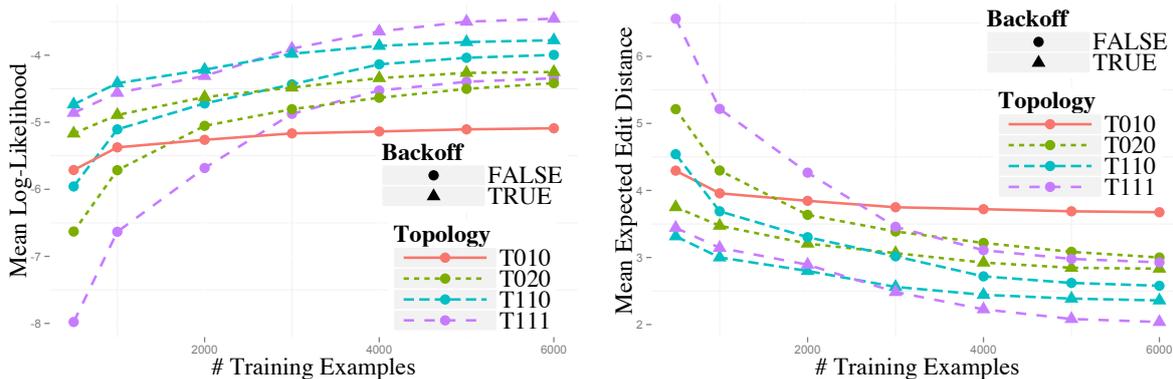


Figure 2: (a) Mean $\log p(y | x)$ for held-out test examples. (b) Mean expected edit distance (similarly).

(Mays et al., 1991; Church and Gale, 1991; Kuchic, 1992; Andrews et al., 2014).

To eliminate experimental confounds, we use no dictionary or language model as one would in practice, but directly evaluate our ability to model $p(\text{correct} | \text{misspelled})$. Consider $(x_k, y_k) = (\text{feel}, \text{feel})$. Our model defines $p(y | x_k)$ for all y . Our training objective (section 6) tries to make this large for $y = y_k$. A *contextual* edit model learns here that $e \mapsto \epsilon$ is more likely in the context of ee .

We report on test data how much probability mass lands on the true y_k . We also report how much mass lands “near” y_k , by measuring the expected edit distance of the predicted y to the truth. Expected edit distance is defined as $\sum_y p_\theta(y | x_k) d(y, y_k)$ where $d(y, y_k)$ is the Levenshtein distance between two strings. It can be computed using standard finite-state algorithms (Mohri, 2003).

8.1 Data

We use an annotated corpus (Aramaki, 2010) of 50000 misspelled words x from tweets along with their corrections y . All examples have $d(x, y) = 1$ though we do not exploit this fact. We randomly selected 6000 training pairs and 100 test pairs. We regularized the objective by adding $\lambda \cdot \|\theta\|_2^2$, where for each training condition, we chose λ by coarse grid search to maximize the conditional likelihood of 100 additional development pairs.

8.2 Context Windows and Edit Features

We considered four different settings for the context window sizes (N_1, N_2, N_3) : $(0, 1, 0)$ =stochastic edit distance, $(1, 1, 0)$, $(0, 2, 0)$, and $(1, 1, 1)$.

Our log-linear edit model (section 6) includes a dedicated indicator feature for each contextual edit (C, e) , allowing us to fit *any* conditional distribution $p(e | C)$. In our “backoff” setting, each (C, e) also has 13 binary backoff features that it

shares with other (C', e') . So we have a total of 14 feature templates, which generate over a million features in our largest model. The shared features let us learn that certain *properties* of a contextual edit tend to raise or lower its probability (and the regularizer encourages such generalization).

Each contextual edit (C, e) can be characterized as a 5-tuple (s, t, C_1, C'_2, C_3) : it replaces $s \in \Sigma_x \cup \{\epsilon\}$ with $t \in \Sigma_y \cup \{\epsilon\}$ when s falls between C_1 and C'_2 (so $C_2 = sC'_2$) and t is preceded by C_3 . Then each of the 14 features of (C, e) indicates that a particular subset of this 5-tuple has a particular value. The subset always includes s, t , or both. It never includes C_1 or C'_2 without s , and never includes C_3 without t .

8.3 Results

Figures 2a and 2b show the learning curves. We see that both metrics improve with more training data; with more context; and with backoff. With backoff, all of the contextual edit models substantially beat ordinary stochastic edit distance, and their advantage grows with training size.

9 Conclusion

We have presented a trainable, featurizable model of *contextual* edit distance. Our main contribution is an efficient encoding of such a model as a tight PFST—that is, a WFST that is guaranteed to directly define conditional string probabilities without need for further normalization. We are releasing OpenFST-compatible code that can train both PFSTs and WFSTs (Cotterell and Renduchintala, 2014). We formally defined PFSTs, described their speed advantage at training time, and noted that they are crucial in settings where the input string is unknown. In future, we plan to deploy our PFSTs in such settings.

References

- Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. 2007. OpenFst: A general and efficient weighted finite-state transducer library. In *Implementation and Application of Automata*, pages 11–23. Springer.
- Nicholas Andrews, Jason Eisner, and Mark Dredze. 2014. Robust entity clustering via phylogenetic inference. In *Proceedings of ACL*.
- Eiji Aramaki. 2010. Typo corpus. Available at <http://luululu.com/tweet/#cr>, January.
- Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Proceedings of HLT-NAACL*, pages 582–590.
- Alexandre Bouchard-Côté, Percy Liang, Thomas L. Griffiths, and Dan Klein. 2007. A probabilistic approach to language change. In *NIPS*.
- Colin Cherry and Dekang Lin. 2003. A probability model to improve word alignment. In *Proceedings of ACL*, pages 88–95.
- Zhiyi Chi and Stuart Geman. 1998. Estimation of probabilistic context-free grammars. *Computational Linguistics*, 24(2):299–305.
- Kenneth W. Church and William A. Gale. 1991. Probability scoring for spelling correction. *Statistics and Computing*, 1(2):93–103.
- Ryan Cotterell and Adithya Renduchintala. 2014. brezel: A library for training FSTs. Technical report, Johns Hopkins University.
- Markus Dreyer, Jason R. Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *Proceedings of EMNLP, EMNLP '08*, pages 1080–1089.
- Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM Model 2. In *Proceedings of NAACL-HLT*, pages 644–648.
- Jason Eisner. 2001. Expectation semirings: Flexible EM for learning finite-state transducers. In *Proceedings of the ESSLLI Workshop on Finite-State Methods in NLP*.
- Dale Gerdemann and Gertjan van Noord. 1999. Transducers from rewrite rules with backreferences. In *Proceedings of EACL*.
- Ronald M. Kaplan and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–378.
- Dan Klein and Christopher D. Manning. 2002. Conditional structure versus conditional estimation in NLP models. In *Proceedings of EMNLP*, pages 9–16.
- Kevin Knight and Jonathan Graehl. 2005. An overview of probabilistic tree transducers for natural language processing. In *Proc. of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*.
- Karen Kukich. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys (CSUR)*, 24(4):377–439.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*.
- Yang Liu, Qun Liu, and Shouxun Lin. 2005. Log-linear models for word alignment. In *Proceedings of ACL*, pages 459–466.
- Eric Mays, Fred J. Damerau, and Robert L. Mercer. 1991. Context based spelling correction. *Information Processing & Management*, 27(5):517–522.
- Andrew McCallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of ICML*.
- Mehryar Mohri and Richard Sproat. 1996. An efficient compiler for weighted rewrite rules. In *Proceedings of ACL*, pages 231–238.
- Mehryar Mohri. 1997. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311.
- Mehryar Mohri. 2003. Edit-distance of weighted automata: General definitions and algorithms. *International Journal of Foundations of Computer Science*, 14(06):957–982.
- Eric Sven Ristad and Peter N. Yianilos. 1998. Learning string edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):522–532.

Labelling Topics using Unsupervised Graph-based Methods

Nikolaos Aletras and Mark Stevenson

Department of Computer Science

University of Sheffield

Regent Court, 211 Portobello

Sheffield, S1 4DP

United Kingdom

{n.aletras, m.stevenson}@dcs.shef.ac.uk

Abstract

This paper introduces an unsupervised graph-based method that selects textual labels for automatically generated topics. Our approach uses the topic keywords to query a search engine and generate a graph from the words contained in the results. PageRank is then used to weigh the words in the graph and score the candidate labels. The state-of-the-art method for this task is supervised (Lau et al., 2011). Evaluation on a standard data set shows that the performance of our approach is consistently superior to previously reported methods.

1 Introduction

Topic models (Hofmann, 1999; Blei et al., 2003) have proved to be a useful way to represent the content of document collections, e.g. (Chaney and Blei, 2012; Ganguly et al., 2013; Gretarsson et al., 2012; Hinneburg et al., 2012; Snyder et al., 2013). In these interfaces, topics need to be presented to users in an easily interpretable way. A common way to represent topics is as set of keywords generated from the n terms with the highest marginal probabilities. For example, a topic about the global financial crisis could be represented by its top 10 most probable terms: FINANCIAL, BANK, MARKET, GOVERNMENT, MORTGAGE, BAILOUT, BILLION, STREET, WALL, CRISIS. But interpreting such lists is not always straightforward, particularly since background knowledge may be required (Chang et al., 2009).

Textual labels could assist with the interpretations of topics and researchers have developed methods to generate these automatically (Mei et al., 2007; Lau et al., 2010; Lau et al., 2011). For example, a topic which has keywords SCHOOL, STUDENT, UNIVERSITY, COLLEGE, TEACHER, CLASS, EDUCATION, LEARN, HIGH, PROGRAM,

could be labelled as EDUCATION and a suitable label for the topic shown above would be GLOBAL FINANCIAL CRISIS. Approaches that make use of alternative modalities, such as images (Aletras and Stevenson, 2013), have also been proposed.

Mei et al. (2007) label topics using statistically significant bigrams identified in a reference collection. Magatti et al. (2009) introduced an approach for labelling topics that relied on two hierarchical knowledge resources labelled by humans, while Lau et al. (2010) proposed selecting the most representative word from a topic as its label. Hulpus et al. (2013) make use of structured data from Wikipedia to label topics.

Lau et al. (2011) proposed a method for automatically labelling topics using information from Wikipedia. A set of candidate labels is generated from Wikipedia article titles by querying using topic terms. Additional labels are then generated by chunk parsing the article titles to identify n -grams that represent Wikipedia articles as well. Outlier labels (less relevant to the topic) are identified and removed. Finally, the top-5 topic terms are added to the candidate set. The labels are ranked using Support Vector Regression (SVR) (Vapnik, 1998) and features extracted using word association measures (i.e. PMI, t -test, χ^2 and Dice coefficient), lexical features and search engine ranking. Lau et al. (2011) report two versions of their approach, one unsupervised (which is used as a baseline) and another which is supervised. They reported that the supervised version achieves better performance than a previously reported approach (Mei et al., 2007).

This paper introduces an alternative graph-based approach which is unsupervised and less computationally intensive than Lau et al. (2011). Our method uses topic keywords to form a query. A graph is generated from the words contained in the search results and these are then ranked using the PageRank algorithm (Page et al., 1999; Mihal-

```
{'Description': 'Microsoft will accelerate your journey to cloud computing with an agile and responsive datacenter built from your existing technology investments.',
'DisplayUrl': 'www.microsoft.com/en-us/server-cloud/datacenter/virtualization.aspx',
'ID': 'a42b0908-174e-4f25-b59c-70bdf394a9da',
'Title': 'Microsoft | Server & Cloud | Datacenter | Virtualization ...',
'Url': 'http://www.microsoft.com/en-us/server-cloud/datacenter/virtualization.aspx',
... }
```

Figure 1: Sample of the metadata associated with a search result.

cea and Tarau, 2004). Evaluation on a standard data set shows that our method consistently outperforms the best performing previously reported method, which is supervised (Lau et al., 2011).

2 Methodology

We use the topic keywords to query a search engine. We assume that the search results returned are relevant to the topic and can be used to identify and weigh relevant keywords. The most important keywords can be used to generate keyphrases for labelling the topic or weight pre-existing candidate labels.

2.1 Retrieving and Processing Text Information

We use the approach described by Lau et al. (2011) to generate candidate labels from Wikipedia articles. The 10 terms with the highest marginal probabilities in the topic are used to query Wikipedia and the titles of the articles retrieved used as candidate labels. Further candidate labels are generated by processing the titles of these articles to identify noun chunks and n-grams within the noun chunks that are themselves the titles of Wikipedia articles. Outlier labels, identified using a similarity measure (Grieser et al., 2011), are removed. This method has been proved to produce labels which effectively summarise a topic’s main subject.

However, it should be noted that our method is flexible and could be applied to any set of candidate labels. We have experimented with various approaches to candidate label generation but chose to report results using the approach described by Lau et al. (2011) to allow direct comparison of approaches.

Information obtained from web searches is used to identify the best labels from the set of candidates. The top n keywords, i.e. those with highest marginal probability within the topic, are used to

form a query which was submitted to the Bing¹ search engine. Textual information included in the Title field² of the search results metadata was extracted. Each title was tokenised using openNLP³ and stop words removed.

Figure 1 shows a sample of the metadata associated with a search result for the topic: VMWARE, SERVER, VIRTUAL, ORACLE, UPDATE, VIRTUALIZATION, APPLICATION, INFRASTRUCTURE, MANAGEMENT, MICROSOFT.

2.2 Creating a Text Graph

We consider any remaining words in the search result metadata as nodes, $v \in V$, in a graph $G = (V, E)$. Each node is connected to its neighbouring words in a context window of $\pm n$ words. In the previous example, the words added to the graph from the Title of the search result are *microsoft, server, cloud, datacenter* and *virtualization*.

We consider both unweighted and weighted graphs. When the graph is unweighted we assume that all the edges have a weight $e = 1$. In addition, we weight the edges of the graph by computing the relatedness between two nodes, v_i and v_j , as their normalised Pointwise Mutual Information (NPMI) (Bouma, 2009). Word co-occurrences are computed using Wikipedia as a reference corpus. Pairs of words are connected with edges only if $NPMI(w_i, w_j) > 0.2$ avoiding connections between words co-occurring by chance and hence introducing noise.

2.3 Identifying Important Terms

Important terms are identified by applying the PageRank algorithm (Page et al., 1999) in a similar way to the approach used by Mihalcea and

¹<http://www.bing.com/>

²We also experimented with using the Description field but found that this reduced performance.

³<http://opennlp.apache.org/>

Tarau (2004) for document keyphrase extraction. The PageRank score (Pr) over G for a word (v_i) can be computed by the following equation:

$$Pr(v_i) = d \cdot \sum_{v_j \in C(v_i)} \frac{sim(v_i, v_j)}{\sum_{v_k \in C(v_j)} sim(v_j, v_k)} Pr(v_j) + (1 - d)\mathbf{v} \quad (1)$$

where $C(v_i)$ denotes the set of vertices which are connected to the vertex v_i . d is the damping factor which is set to the default value of $d = 0.85$ (Page et al., 1999). In standard PageRank all elements of the vector \mathbf{v} are the same, $\frac{1}{N}$ where N is the number of nodes in the graph.

2.4 Ranking Labels

Given a candidate label $L = \{w_1, \dots, w_m\}$ containing m keywords, we compute the score of L by simply adding the PageRank scores of its constituent keywords:

$$Score(L) = \sum_{i=1}^m Pr(w_i) \quad (2)$$

The label with the highest score amongst the set of candidates is selected to represent the topic. We also experimented with normalised versions of the score, e.g. mean of the PageRank scores. However, this has a negative effect on performance since it favoured short labels of one or two words which were not sufficiently descriptive of the topics. In addition, we expect that candidate labels containing words that do not appear in the graph (with the exception of stop words) are unlikely to be good labels for the topic. In these cases the score of the candidate label is set to 0. We also experimented with removing this restriction but found that it lowered performance.

3 Experimental Evaluation

3.1 Data

We evaluate our method on the publicly available data set published by Lau et al. (2011). The data set consists of 228 topics generated using text documents from four domains, i.e. blog posts (**BLOGS**), books (**BOOKS**), news articles (**NEWS**) and scientific articles from the biomedical domain (**PUBMED**). Each topic is represented by its ten most probable keywords. It is also associated with candidate labels and human ratings

denoting the appropriateness of a label given the topic. The full data set consists of approximately 6,000 candidate labels (27 labels per topic).

3.2 Evaluation Metrics

Our evaluation follows the framework proposed by Lau et al. (2011) using two metrics, i.e. **Top-1 average rating** and **nDCG**, to compare various labelling methods.

Top-1 average rating is the average human rating (between 0 and 3) assigned to the top-ranked label proposed by the system. This provides an indication of the overall quality of the label the system judges as the best one.

Normalised discounted cumulative gain (**nDCG**) (Järvelin and Kekäläinen, 2002; Croft et al., 2009) compares the label ranking proposed by the system to the ranking provided by human annotators. The discounted cumulative gain at position p , DCG_p , is computed using the following equation:

$$DCG_p = rel_1 + \sum_{i=2}^p \frac{rel_i}{\log_2(i)} \quad (3)$$

where rel_i is the relevance of the label to the topic in position i . Then nDCG is computed as:

$$nDCG_p = \frac{DCG_p}{IDCG_p} \quad (4)$$

where $IDCG_p$ is the supervised ranking of the image labels, in our experiments this is the ranking provided by the scores in the human annotated data set.

3.3 Model Parameters

Our proposed model requires two parameters to be set: the context window size when connecting neighbouring words in the graph and the number of the search results considered when constructing the graph.

We experimented with different sizes of context window, n , between ± 1 words to the left and right and all words in the title. The best results were obtained when $n = 2$ for all of the domains. In addition, we experimented with varying the number of search results between 10 and 300. We observed no noticeable difference in the performance when the number of search results is equal or greater than 30 (see below). We choose to report results obtained using 30 search results for each topic. Including more results did not improve performance but required additional processing.

Domain	Model	Top-1 Av. Rating	nDCG-1	nDCG-3	nDCG-5
BLOGS	Lau et al. (2011)-U	1.84	0.75	0.77	0.79
	Lau et al. (2011)-S	1.98	0.81	0.82	0.83
	PR	2.05†	0.83	0.84	0.83
	PR-NPMI	2.08†	0.84	0.84	0.83
	Upper bound	2.45	1.00	1.00	1.00
BOOKS	Lau et al. (2011)-U	1.75	0.77	0.77	0.79
	Lau et al. (2011)-S	1.91	0.84	0.81	0.83
	PR	1.98†	0.86	0.88	0.87
	PR-NPMI	2.01†	0.87	0.88	0.87
	Upper bound	2.29	1.00	1.00	1.00
NEWS	Lau et al. (2011)-U	1.96	0.80	0.79	0.78
	Lau et al. (2011)-S	2.02	0.82	0.82	0.84
	PR	2.04†	0.83	0.81	0.81
	PR-NPMI	2.05†	0.83	0.81	0.81
	Upper bound	2.45	1.00	1.00	1.00
PUBMED	Lau et al. (2011)-U	1.73	0.75	0.77	0.79
	Lau et al. (2011)-S	1.79	0.77	0.82	0.84
	PR	1.88†‡	0.80	0.80	0.80
	PR-NPMI	1.90†‡	0.81	0.80	0.80
	Upper bound	2.31	1.00	1.00	1.00

Table 1: Results for Various Approaches to Topic Labelling (†: significant difference (t-test, $p < 0.05$) to Lau et al. (2011)-U; ‡: significant difference ($p < 0.05$) to Lau et al. (2011)-S).

4 Results and Discussion

Results are shown in Table 1. Performance when PageRank is applied to the unweighted (**PR**) and NPMI-weighted graphs (**PR-NPMI**) (see Section 2.2) is shown. Performance of the best unsupervised (**Lau et al. (2011)-U**) and supervised (**Lau et al. (2011)-S**) methods reported by Lau et al. (2011) are shown. Lau et al. (2011)-U uses the average χ^2 scores between the topic keywords and the label keywords while Lau et al. (2011)-S uses SVR to combine evidence from all features. In addition, upper bound figures, the maximum possible value given the scores assigned by the annotators, are also shown.

The results obtained by applying PageRank over the unweighted graph (2.05, 1.98, 2.04 and 1.88) are consistently better than the supervised and unsupervised methods reported by Lau et al. (2011) for the Top-1 Average scores and this improvement is observed in all domains. The difference is significant (t-test, $p < 0.05$) for the unsupervised method. A slight improvement in per-

formance is observed when the weighted graph is used (2.08, 2.01, 2.05 and 1.90). This is expected since the weighted graph contains additional information about word relatedness. For example, the word *hardware* is more related and, therefore, closer in the graph to the word *virtualization* than to the word *investments*.

Results from the nDCG metric imply that our methods provide better rankings of the candidate labels in the majority of the cases. It is outperformed by the best supervised approach in two domains, NEWS and PUBMED, using the nDCG-3 and nDCG-5 metrics. However, the best label proposed by our methods is judged to be better (as shown by the nDCG-1 and Top-1 Av. Rating scores), demonstrating that it is only the lower ranked labels in our approach that are not as good as the supervised approach.

An interesting finding is that, although limited in length, the textual information in the search result’s metadata contain enough salient terms relevant to the topic to provide reliable estimates of

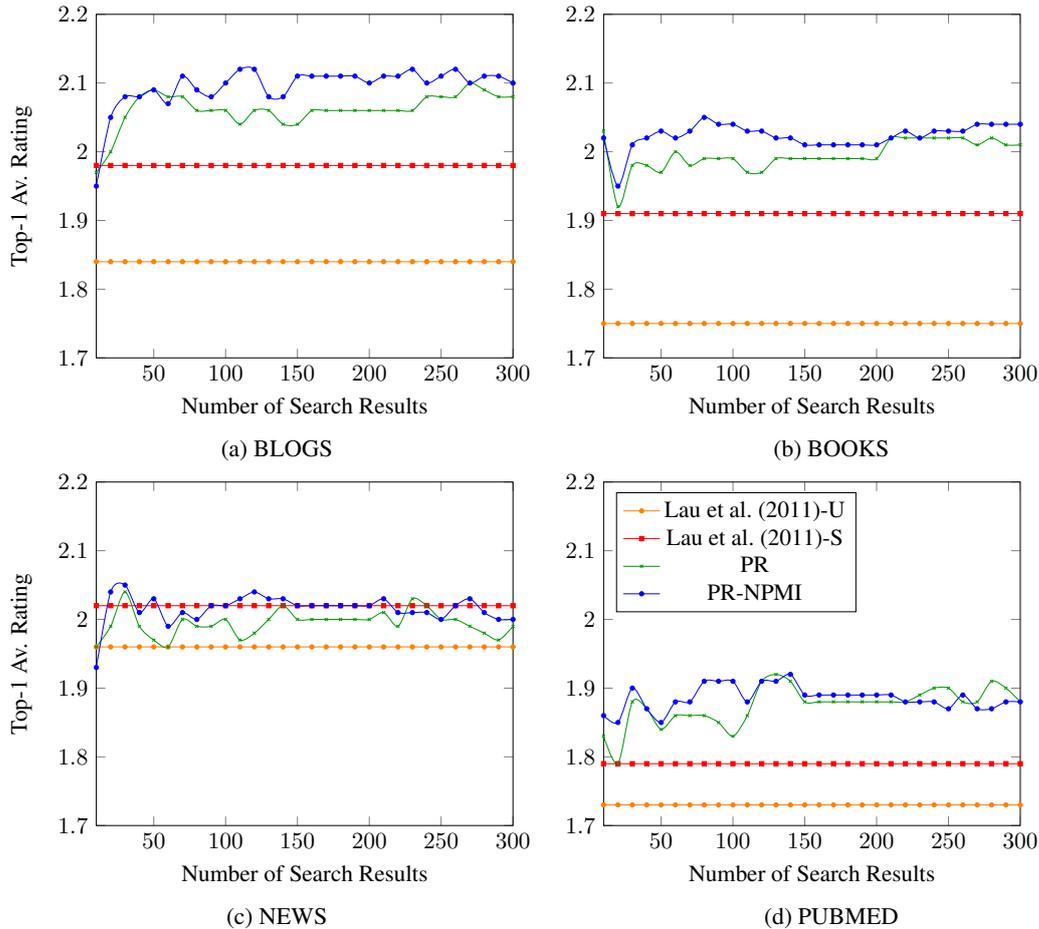


Figure 2: Top-1 Average Rating obtained for different number of search results.

term importance. Consequently, it is not necessary to measure semantic similarity between topic keywords and candidate labels as previous approaches have done. In addition, performance improvement gained from using the weighted graph is modest, suggesting that the computation of association scores over a large reference corpus could be omitted if resources are limited.

In Figure 2, we show the scores of Top-1 average rating obtained in the different domains by experimenting with the number of search results used to generate the text graph. The most interesting finding is that performance is stable when 30 or more search results are considered. In addition, we observe that quality of the topic labels in the four domains remains stable, and higher than the supervised method, when the number of search results used is between 150 and 200. The only domain in which performance of the supervised method is sometimes better than the approach proposed here is NEWS. The main reason is that news topics are more fine grained and the candidate

labels of better quality (Lau et al., 2011) which has direct impact in good performance of ranking methods.

5 Conclusion

We described an unsupervised graph-based method to associate textual labels with automatically generated topics. Our approach uses results retrieved from a search engine using the topic keywords as a query. A graph is generated from the words contained in the search results metadata and candidate labels ranked using the PageRank algorithm. Evaluation on a standard data set shows that our method consistently outperforms the supervised state-of-the-art method for the task.

Acknowledgments

We would like to thank Jey Han Lau for providing us with the labels selected by Lau et al. (2011)-U and Lau et al. (2011)-S. We also thank Daniel Preoțiuc-Pietro for his useful comments on early drafts of this paper.

References

- Nikolaos Aletras and Mark Stevenson. 2013. Representing topics using images. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 158–167, Atlanta, Georgia.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.
- Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. In *Proceedings of GSCL*.
- Allison June-Barlow Chaney and David M. Blei. 2012. Visualizing topic models. In *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media*, Dublin, Ireland.
- Jonathan Chang, Jordan Boyd-Graber, and Sean Gerish. 2009. Reading Tea Leaves: How Humans Interpret Topic Models. *Neural Information*, pages 1–9.
- Bruce W. Croft, Donald Metzler, and Trevor Strohman. 2009. *Search engines: Information retrieval in practice*. Addison-Wesley.
- Debasis Ganguly, Manisha Ganguly, Johannes Leveling, and Gareth J.F. Jones. 2013. TopicVis: A GUI for Topic-based feedback and navigation. In *Proceedings of the Thirty-Sixth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 13)*, Dublin, Ireland.
- Brynjar Gretarsson, John O’Donovan, Svetlin Bostandjiev, Tobias Höllerer, Arthur Asuncion, David Newman, and Padhraic Smyth. 2012. TopicNets: Visual analysis of large text corpora with topic modeling. *ACM Trans. Intell. Syst. Technol.*, 3(2):23:1–23:26.
- Karl Grieser, Timothy Baldwin, Fabian Bohnert, and Liz Sonenberg. 2011. Using Ontological and Document Similarity to Estimate Museum Exhibit Relatedness. *Journal on Computing and Cultural Heritage (JOCCH)*, 3(3):10:1–10:20.
- Alexander Hinneburg, Rico Preiss, and René Schröder. 2012. TopicExplorer: Exploring document collections with topic models. In Peter A. Flach, Tijl Bie, and Nello Cristianini, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 7524 of *Lecture Notes in Computer Science*, pages 838–841. Springer Berlin Heidelberg.
- Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR ’99)*, pages 50–57, Berkeley, California, United States.
- Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. 2013. Unsupervised graph-based topic labelling using DBpedia. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining (WSDM ’13)*, pages 465–474, Rome, Italy.
- Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.
- Jey Han Lau, David Newman, Sarvnaz Karimi, and Timothy Baldwin. 2010. Best topic word selection for topic labelling. In *The 23rd International Conference on Computational Linguistics (COLING ’10)*, pages 605–613, Beijing, China.
- Jey Han Lau, Karl Grieser, David Newman, and Timothy Baldwin. 2011. Automatic labelling of topic models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1536–1545, Portland, Oregon, USA.
- Davide Magatti, Silvia Calegari, Davide Ciucci, and Fabio Stella. 2009. Automatic Labeling of Topics. In *Proceedings of the 9th International Conference on Intelligent Systems Design and Applications (ICSDA ’09)*, pages 1227–1232, Pisa, Italy.
- Qiaozhu Mei, Xuehua Shen, and Cheng Xiang Zhai. 2007. Automatic Labeling of Multinomial Topic Models. In *Proceedings of the 13th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD ’07)*, pages 490–499, San Jose, California.
- Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into texts. In *Proceedings of International Conference on Empirical Methods in Natural Language Processing (EMNLP ’04)*, pages 404–411, Barcelona, Spain.
- Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab.
- Justin Snyder, Rebecca Knowles, Mark Dredze, Matthew Gormley, and Travis Wolfe. 2013. Topic models and metadata for visualizing text corpora. In *Proceedings of the 2013 NAACL-HLT Demonstration Session*, pages 5–9, Atlanta, Georgia. Association for Computational Linguistics.
- Vladimir N Vapnik. 1998. *Statistical learning theory*. Wiley, New York.

Training a Korean SRL System with Rich Morphological Features

Young-Bum Kim, Heemoon Chae, Benjamin Snyder and Yu-Seop Kim*

University of Wisconsin-Madison, Hallym University*

{ybkim, hmchae21, bsnyder}@cs.wisc.edu, yskim01@hallym.ac.kr*

Abstract

In this paper we introduce a semantic role labeler for Korean, an agglutinative language with rich morphology. First, we create a novel training source by semantically annotating a Korean corpus containing fine-grained morphological and syntactic information. We then develop a supervised SRL model by leveraging morphological features of Korean that tend to correspond with semantic roles. Our model also employs a variety of latent morpheme representations induced from a larger body of unannotated Korean text. These elements lead to state-of-the-art performance of 81.07% labeled F1, representing the best SRL performance reported to date for an agglutinative language.

1 Introduction

Semantic Role Labeling (SRL) is the task of automatically annotating the predicate-argument structure in a sentence with semantic roles. Ever since Gildea and Jurafsky (2002), SRL has become an important technology used in applications requiring semantic interpretation, ranging from information extraction (Frank et al., 2007) and question answering (Narayanan and Harabagiu, 2004), to practical problems including textual entailment (Burchardt et al., 2007) and pictorial communication systems (Goldberg et al., 2008).

SRL systems in many languages have been developed as the necessary linguistic resources become available (Taulé et al., 2008; Xue and Palmer, 2009; Böhmová et al., 2003; Kawahara et al., 2002). Seven languages were the subject of the CoNLL-2009 shared task in syntactic and semantic parsing (Hajič et al., 2009). These languages can be categorized into three broad morphological types: fusional (4), analytic (2), and one agglutinative language.

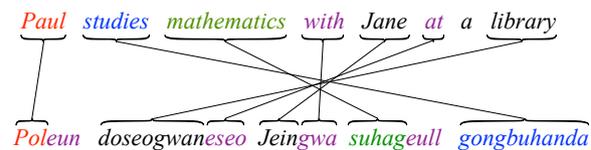


Figure 1: English (SVO) and Korean (SOV) words alignment. The subject, verb, and object are highlighted as red, blue, and green, respectively. Also, prepositions and suffixes are highlighted as purple.

Björkelund et al. (2009) report an average labeled semantic F1-score of 80.80% across these languages. The highest performance was achieved for the analytic language group (82.12%), while the agglutinative language, Japanese, yielded the lowest performance (76.30%). Agglutinative languages such as Japanese, Korean, and Turkish are computationally difficult due to word-form sparsity, variable word order, and the challenge of using rich morphological features.

In this paper, we describe a Korean SRL system which achieves 81% labeled semantic F1-score. As far as we know, this is the highest accuracy obtained for Korean, as well as any agglutinative language. Figure 1 displays a English/Korean sentence pair, highlighting the SOV word order of Korean as well as its rich morphological structure. Two factors proved crucial in the performance of our SRL system: (i) The analysis of fine-grained morphological tags specific to Korean, and (ii) the use of latent stem and morpheme representations to deal with sparsity. We incorporated both of these elements in a CRF (Lafferty et al., 2001) role labeling model.

Besides the contribution of this model and SRL system, we also report on the creation and availability of a new semantically annotated Korean corpus, covering over 8,000 sentences. We used this corpus to develop, train, and test our Korean SRL model. In the next section, we describe the process of corpus creation in more detail.

2 A Semantically Annotated Korean Corpus

We annotated predicate-argument structure of verbs in a corpus from the Electronics and Telecommunications Research Institute of Korea (ETRI).¹ Our corpus was developed over two years using a specialized annotation tool (Song et al., 2012), resulting in more than 8,000 semantically annotated sentences. As much as possible, annotations followed the PropBank guidelines for English (Bonial et al., 2010).

We view our work as building on the efforts of the Penn Korean PropBank (PKPB).² Our corpus is roughly similar in size to the PKPB, and taken together, the two Korean corpora now total about half the size of the Penn English PropBank. One advantage of our corpus is that it is built on top of the ETRI Korean corpus, which uses a richer Korean morphological tagging scheme than the Penn Korean Treebank. Our experiments will show that these finer-grained tags are crucial for achieving high SRL accuracy.

All annotations were performed by two people working in a team. At first, each annotator assigns semantic roles independently and then they discuss to reduce disagreement of their annotation results. Initially, the disagreement rate between two annotators was about 14%. After 4 months of this process, the disagreement rate fell to 4% through the process of building annotation rules for Korean. The underlying ETRI syntactically-annotated corpus contains the dependency tree structure of sentences with morpho-syntactic tags. It includes 101,602 multiple-clause sentences with 21.66 words on average.

We encountered two major difficulties during annotation. First, the existing Korean frame files from the Penn Korean PropBank include 2,749 verbs, covering only 13.87% of all the verbs in the ETRI corpus. Secondly, no Korean PropBanking guidelines have previously been published, leading to uncertainty in the initial stages of annotation. These uncertainties were gradually resolved through the iterative process of resolving inter-annotator disagreements.

Table 1 shows the semantic roles considered in our annotated corpus. Although these are based on the general English PropBank guidelines (Bonial et al., 2010), they also differ in that we used only

Roles	Definition	Rate
ARG0	Agent	10.02%
ARG1	Patient	26.73%
ARG2	Start point / Benefactive	5.18%
ARG3	Ending point	1.10%
ARGM-ADV	Adverbial	1.26%
ARGM-CAU	Cause	1.17%
ARGM-CND	Condition	0.36%
ARGM-DIR	Direction	0.35%
ARGM-DIS	Discourse	28.71%
ARGM-EXT	Extent	4.50%
ARGM-INS	Instrument	1.04%
ARGM-LOC	Locative	4.51%
ARGM-MNR	Manner	8.72%
ARGM-NEG	Negation	0.26%
ARGM-PRD	Predication	0.27%
ARGM-PRP	Purpose	0.77%
ARGM-TMP	Temporal	5.05%

Table 1: Semantic roles in our annotated corpus.

4 numbered arguments from ARG0 to ARG3 instead of 5 numbered arguments. We thus consider 17 semantic roles in total. Four of them are numbered roles, describing the essential arguments of a predicate. The other roles are called modifier roles that play more of an adjunct role.

We have annotated semantic roles by following the PropBank annotation guideline (Bonial et al., 2010) and by using frame files of the Penn Korean PropBank built by Palmer et al. (2006). The PropBank and our corpus are not exactly compatible, because the former is built on constituency-based parse trees, whereas our corpus uses dependency parses.

More importantly, the tagsets of these corpora are not fully compatible. The PKPB uses much coarser morpho-syntactic tags than the ETRI corpus. For example, the PCA tag in PKPB used for a case suffix covers four different functioning tags used in our corpus. Using coarser suffix tags can seriously degrade SRL performance, as we show in Section 6, where we compare the performance of our model on both the new corpus and the older PKPB.

¹http://voice.etri.re.kr/db/db_pop.asp?code=88

²<http://catalog.ldc.upenn.edu/LDC2006T03>

3 Previous Work

Korean SRL research has been limited to domestically published Korean research on small corpora. Therefore, the most direct precedent to the present work is a section in Björkelund et al. (2009) on Japanese SRL. They build a classifier consisting of 3 stages: predicate disambiguation, argument identification, and argument classification.

They use an L_2 -regularized linear logistic regression model cascaded through these three stages, achieving F1-score of 80.80% on average for 7 languages (Catalan, Chinese, Czech, English, German, Japanese and Spanish). The lowest reported performance is for Japanese, the only agglutinative language in their data set, achieving F1-score of 76.30%. This result showcases the computational difficulty of dealing with morphologically rich agglutinative languages. As we discuss in Section 5, we utilize these same features, but also add a set of Korean-specific features to capture aspects of Korean morphology.

Besides these morphological features, we also employ latent continuous and discrete morpheme representations induced from a larger body of unannotated Korean text. As our experiments below show, these features improve performance by dealing with sparsity issues. Such features have been useful in a variety of English NLP models, including chunking, named entity recognition (Turian et al., 2010), and spoken language understanding (Anastasakos et al., 2014). Unlike the English models, we use individual morphemes as our unit of analysis.

4 Model

For the semantic role task, the input is a sentence consisting of a sequence of words $x = x_1, \dots, x_n$ and the output is a sequence of corresponding semantic tags $y = y_1, \dots, y_n$. Each word consists of a stem and some number of suffix morphemes, and the semantic tags are drawn from the set $\{\text{NONE}, \text{ARGO}, \dots, \text{ARGM-TMP}\}$. We model the conditional probability $p(y|x)$ using a CRF model:

$$Z(x)^{-1} \prod_{i=1}^x \exp \sum_m \lambda_m f_m(y_{i-1}, y_i, x, i),$$

where $f_m(y_{i-1}, y_i, x, i)$ are the feature functions. These feature functions include transition features

that identify the tag bigram (y_{i-1}, y_i) , and emission features that combine the current semantic tag (y_i) with instantiated feature templates extracted from the sentence x and its underlying morphological and dependency analysis. The function Z is the normalizing function, which ensures that $p(y|x)$ is a valid probability distribution. We used 100 iteration of averaged perceptron algorithm to train the CRF.

5 Features

We detail the feature templates used for our experiments in Table 2. These features are categorized as either general features, Korean-specific features, or latent morpheme representation features. Korean-specific features are built upon the morphological analysis of the suffix agglutination of the current word x_i .

Korean suffixes are traditionally classified into two groups called *Josa* and *Eomi*. *Josa* is used to define nominal cases and modify other phrases, while *Eomi* is an ending of a verb or an adjective to define a tense, show an attitude, and connect or terminate a sentence. Thus, the *Eomi* and *Josa* categorization plays an important role in signaling semantic roles. Considering the functions of *Josa* and *Eomi*, we expect that numbered roles are relevant to *Josa* while modifier roles are more closely related to *Eomi*. The one exception is adverbial *Josa*, making the attached phrase an adverb that modifies a verb predicate.

For all feature templates, “A-” or “P-” are used respectively to signify that the feature corresponds to the argument in question (x_i), or rather is derived from the verbal predicate that the argument depends on.

General features: We use and modify 18 features used for Japanese from the prior work of Björkelund et al. (2009), excluding SENSE, POSITION, and re-ranker features.

- Stem: a stem without any attachment. For instance, the first word *Poleun* at the Figure 1 consists of a stem *Pol* plus *Josa eun*.
- POS.Lv1: the first level (coarse classification) of a POS tag such as noun, verb, adjective, or adverb.

Feature	Description
A-Stem, P-Stem	Stem of an argument and a predicate
A-POS_Lv1, P-POS_Lv1	Coarse-grained POS of A-Stem and P-Stem
A-POS_Lv2, P-POS_Lv2	Fine-grained POS of A-Stem and P-Stem
A-Case, P-Case	Case of A-Stem and P-Stem
A-LeftmostChildStem	Stem of the leftmost child of an argument
A-LeftSiblingStem	Stem of the left sibling of an argument
A-LeftSiblingPOS_Lv1	Coarse-grained POS of A-LeftSiblingStem
A-LeftSiblingPOS_Lv2	Fine-grained POS of A-LeftSiblingStem
A-RightSiblingPOS_Lv1	Coarse-grained POS of a stem of the right sibling of an argument
A-RightSiblingPOS_Lv2	Fine-grained POS of a stem of the right sibling of an argument
P-ParentStem	Stem of a parent of a predicate
P-ChildStemSet	Set of stems of children of a predicate
P-ChildPOSSet_Lv1	Set of coarse POS of P-ChildStemSet
P-ChildCaseSet	Set of cases of P-childStemSet
A-JosaExist	If 1, Josa exists in an argument, otherwise 0.
A-JosaClass	Linguistic classification of Josa
A-JosaLength	Number of morphemes consisting of Josa
A-JosaMorphemes	Each morpheme consisting of Josa
A-JosaIdentity	Josa of an argument
A-EomiExist	If 1, Eomi exists in an argument, otherwise 0.
A-EomiClass_Lv1	Linguistic classification of Eomi
A-EomiClass_Lv2	Another linguistic classification of Eomi
A-EomiLength	Number of morphemes consisting of Eomi
A-EomiMorphemes	Each morpheme consisting of Eomi
A-EomiIdentity	Eomi of an argument
A-StemRepr	Stem representation of an argument
A-JosaRepr	Josa representation of an argument
A-EomiRepr	Eomi representation of an argument

Table 2: Features used in our SRL experiments. Features are grouped as General, Korean-specific, or Latent Morpheme Representations. For the last group, we employ three different methods to build them: (i) CCA, (ii) deep learning, and (iii) Brown clustering.

- POS_Lv2: the second level (fine classification) of a POS tag. If POS_Lv1 is *noun*, either a proper noun, common noun, or other kinds of nouns is the POS_Lv2.
- Case: the case type such as SBJ, OBJ, or COMP.
- A-JosaLength: the number of morphemes consisting of Josa. At most five morphemes are combined to consist of one Josa in our data set.
- A-JosaMorphemes: Each morpheme composing the Josa.

The above features are also applied to some dependency children, parents, and siblings of arguments as shown in Table 2.

Korean-specific features: We have 11 different kinds of features for the Josa (5) and Eomi (6). We highlight several below:

- A-JosaExist: an indicator feature checking any Josa whether or not exists in an argument. It is set to 1 if any Josa exists, otherwise 0.
- A-JosaClass: the linguistic classification of Josa with a total of 8 classes. These classes are adverbial, auxiliary, complemental, connective, determinative, objective, subjective, and vocative.
- A-JosaIdentity: The Josa itself.
- A-EomiClass_Lv1: the linguistic classification of Eomi with a total of 14 classes. These 14 classes are adverbial, determinative, coordinate, exclamatory, future tense, honorific, imperative, interrogative, modesty, nominal, normal, past tense, petitionary, and subordinate.
- A-EomiClass_Lv2: Another linguistic classification of Eomi with a total of 4 classes. The four classes are closing, connection, prefinal, and transmutation. The EomiClass_Lv1 and Lv2 are combined to display the characteristic of Eomi such as ‘Nominal Transmutation Eomi’, but not all combinations are possible.

Corpus	Gen	Gen+Kor	Gen+Kor+LMR			
			CCA	Deep	Brown	All
PKPB	64.83%	75.17%	75.51%	75.43%	75.55%	75.54%
Our annotated corpus	66.88%	80.33%	80.88%	80.84%	80.77%	81.07%
PKPB + our annotated corpus	64.86%	78.61%	79.32%	79.44%	78.91%	79.20%

Table 3: Experimental F1-score results on every experiment. Abbreviation on features are Gen: general features, Kor: Korean specific features, LMR: latent morpheme representation features.

Latent morpheme representation features: To alleviate the sparsity, a lingering problem in NLP, we employ three kinds of latent morpheme representations induced from a larger body of unsupervised text data. These are (i) linear continuous representation through Canonical Correlation Analysis (Dhillon et al., 2012), (ii) non-linear continuous representation through Deep learning (Collobert and Weston, 2008), and (iii) discrete representation through Brown Clustering (Tatu and Moldovan, 2005).

The first two representations are 50 dimensional continuous vectors for each morpheme, and the latter is a set of 256 clusters over morphemes.

6 Experiments and Results

We categorized our experiments by the scenarios below, and all results are summarized in Table 3. The F1-score results were investigated for each scenario. We randomly divided our data into 90% training and 10% test sets for all scenarios.

For latent morpheme representations, we used the Donga news article corpus.³ The Donga corpus contains 366,636 sentences with 25.09 words on average. The Domain of this corpus covers typical news articles such as health, entertainment, technology, politics, world and others. We ran Kokoma Korean morpheme analyzer⁴ on each sentence of the Donga corpus to divide words into morphemes to build latent morpheme representations.

1st Scenario: We first tested on general features in previous work (2nd column in Table 3). We achieved 64.83% and 66.88% on the PKPB and our corpus. When the both corpora were combined, we had 64.86%.

2nd Scenario: We then added the Korean-specific morphological features to signify its ap-

propriateness in this scenario. These features increased greatly performance improvements (3rd column in Table 3). Although both the PKPB and our corpus had improvements, the improvements were the most notable on our corpus. This is because PKPB POS tags might be too coarse. We achieved 75.17%, 80.33%, and 78.61% on the PKPB, our corpus, and the combined one, respectively.

3rd Scenario: This scenario is to reveal the effects of the different latent morpheme representations (4-6th columns in Table 3). These three representations are from CCA, deep learning, and Brown clustering. The results gave evidences that all representations increased the performance.

4th Scenario: We augmented our model with all kinds of features (the last column in Table 3). We achieved our best F1-score of 81.07% over all scenarios on our corpus.

7 Conclusion

For Korean SRL, we semantically annotated a corpus containing detailed morphological annotation. We then developed a supervised model which leverages Korean-specific features and a variety of latent morpheme representations to help deal with a sparsity problem. Our best model achieved 81.07% in F1-score. In the future, we will continue to build our corpus and look for the way to use unsupervised learning for SRL to apply to another language which does not have available corpus.

Acknowledgments

We thank Na-Rae Han and Asli Celikyilmaz for helpful discussion and feedback. This research was supported by the Basic Science Research Program of the Korean National Research Foundation (NRF), and funded by the Korean Ministry of Education, Science and Technology (2010-0010612).

³<http://www.donga.com>

⁴<http://kkma.snu.ac.kr/>

References

- Tasos Anastasakos, Young-Bum Kim, and Anoop Deoras. 2014. Task specific continuous word representations for mono and multi-lingual spoken language understanding. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.
- Anders Björkelund, Love Hafdel, and Pierre Nugues. 2009. Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 43–48. Association for Computational Linguistics.
- Alena Böhmová, Jan Hajič, Eva Hajičová, and Barbora Hladká. 2003. The prague dependency treebank. In *Trebanks*, pages 103–127. Springer.
- Claire Bonial, Olga Babko-Malaya, Jinho D Choi, Jena Hwang, and Martha Palmer. 2010. Propbank annotation guidelines. *Center for Computational Language and Education Research, CU-Boulder*.
- Aljoscha Burchardt, Nils Reiter, Stefan Thater, and Anette Frank. 2007. A semantic approach to textual entailment: system evaluation and task analysis. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, RTE '07*, pages 10–15, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.
- Paramveer Dhillon, Jordan Rodu, Dean Foster, and Lyle Ungar. 2012. Two step cca: A new spectral method for estimating vector models of words. *arXiv preprint arXiv:1206.6403*.
- Anette Frank, Hans-Ulrich Krieger, Feiyu Xu, Hans Uszkoreit, Berthold Crysmann, Brigitte Jörg, and Ulrich Schäfer. 2007. Question answering from structured knowledge sources. *Journal of Applied Logic*, 5(1):20 – 48. Questions and Answers: Theoretical and Applied Perspectives.
- Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational linguistics*, 28(3):245–288.
- Andrew B Goldberg, Xiaojin Zhu, Charles R Dyer, Mohamed Eldawy, and Lijie Heng. 2008. Easy as abc?: facilitating pictorial communication via semantically enhanced layout. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 119–126. Association for Computational Linguistics.
- Jan Hajič, Massimiliano Ciaramita, Richard Johanson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, et al. 2009. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–18. Association for Computational Linguistics.
- Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a japanese relevance-tagged corpus. In *LREC*.
- John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.
- Srini Narayanan and Sanda Harabagiu. 2004. Question answering based on semantic structures. In *Proceedings of the 20th international conference on Computational Linguistics, COLING '04*, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Martha Palmer, Shijong Ryu, Jinyoung Choi, Sinwon Yoon, and Yeongmi Jeon. 2006. Korean propbank. *Linguistic data consortium*.
- Hye-Jeong Song, Chan-Young Park, Jung-Kuk Lee, Min-Ji Lee, Yoon-Jeong Lee, Jong-Dae Kim, and Yu-Seop Kim. 2012. Construction of korean semantic annotated corpus. In *Computer Applications for Database, Education, and Ubiquitous Computing*, pages 265–271. Springer.
- Marta Tatu and Dan Moldovan. 2005. A semantic approach to recognizing textual entailment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 371–378. Association for Computational Linguistics.
- Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. Ancora: Multilevel annotated corpora for catalan and spanish. In *LREC*.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.
- Nianwen Xue and Martha Palmer. 2009. Adding semantic roles to the chinese treebank. *Natural Language Engineering*, 15(01):143–172.

Semantic Parsing for Single-Relation Question Answering

Wen-tau Yih Xiaodong He Christopher Meek

Microsoft Research

Redmond, WA 98052, USA

{scottyih, xiaohe, meek}@microsoft.com

Abstract

We develop a semantic parsing framework based on semantic similarity for open domain question answering (QA). We focus on single-relation questions and decompose each question into an entity mention and a relation pattern. Using convolutional neural network models, we measure the similarity of entity mentions with entities in the knowledge base (KB) and the similarity of relation patterns and relations in the KB. We score relational triples in the KB using these measures and select the top scoring relational triple to answer the question. When evaluated on an open-domain QA task, our method achieves higher precision across different recall points compared to the previous approach, and can improve F_1 by 7 points.

1 Introduction

Open-domain question answering (QA) is an important and yet challenging problem that remains largely unsolved. In this paper, we focus on answering single-relation factual questions, which are the most common type of question observed in various community QA sites (Fader et al., 2013), as well as in search query logs. We assumed such questions are answerable by issuing a single-relation query that consists of the relation and an argument entity, against a knowledge base (KB). Example questions of this type include: “*Who is the CEO of Tesla?*” and “*Who founded Paypal?*”

While single-relation questions are easier to handle than questions with more complex and multiple relations, such as “*When was the child of the former Secretary of State in Obama’s administration born?*”, single-relation questions are still far from completely solved. Even in this restricted domain there are a large number of paraphrases of

the same question. That is to say that the problem of mapping from a question to a particular relation and entity in the KB is non-trivial.

In this paper, we propose a semantic parsing framework tailored to single-relation questions. At the core of our approach is a novel semantic similarity model using convolutional neural networks. Leveraging the question paraphrase data mined from the WikiAnswers corpus by Fader et al. (2013), we train two semantic similarity models: one links a mention from the question to an entity in the KB and the other maps a relation pattern to a relation. The answer to the question can thus be derived by finding the relation–entity triple $r(e_1, e_2)$ in the KB and returning the entity not mentioned in the question. By using a general semantic similarity model to match patterns and relations, as well as mentions and entities, our system outperforms the existing rule learning system, PARALEX (Fader et al., 2013), with higher precision at all the recall points when answering the questions in the same test set. The highest achievable F_1 score of our system is 0.61, versus 0.54 of PARALEX.

The rest of the paper is structured as follows. We first survey related work in Sec. 2, followed by the problem definition and the high-level description of our approach in Sec. 3. Sec. 4 details our semantic models and Sec. 5 shows the experimental results. Finally, Sec. 6 concludes the paper.

2 Related Work

Semantic parsing of questions, which maps natural language questions to database queries, is a critical component for KB-supported QA. An early example of this research is the semantic parser for answering geography-related questions, learned using inductive logic programming (Zelle and Mooney, 1996). Research in this line originally used small, domain-specific databases, such as GeoQuery (Tang and Mooney, 2001; Liang et

al., 2013). Very recently, researchers have started developing semantic parsers for large, general-domain knowledge bases like Freebase and DBpedia (Cai and Yates, 2013; Berant et al., 2013; Kwiatkowski et al., 2013). Despite significant progress, the problem remains challenging. Most methods have not yet been scaled to large KBs that can support general open-domain QA. In contrast, Fader et al. (2013) proposed the PARALEX system, which targets answering single-relation questions using an automatically created knowledge base, ReVerb (Fader et al., 2011). By applying simple seed templates to the KB and by leveraging community-authored paraphrases of questions from WikiAnswers, they successfully demonstrated a high-quality lexicon of pattern-matching rules can be learned for this restricted form of semantic parsing.

The other line of work related to our approach is continuous representations for semantic similarity, which has a long history and is still an active research topic. In information retrieval, TF-IDF vectors (Salton and McGill, 1983), latent semantic analysis (Deerwester et al., 1990) and topic models (Blei et al., 2003) take the bag-of-words approach, which captures well the contextual information for documents, but is often too coarse-grained to be effective for sentences. In a separate line of research, deep learning based techniques have been proposed for semantic understanding (Mesnil et al., 2013; Huang et al., 2013; Shen et al., 2014b; Salakhutdinov and Hinton, 2009; Tur et al., 2012). We adapt the work of (Huang et al., 2013; Shen et al., 2014b) for measuring the semantic distance between a question and relational triples in the KB as the core component of our semantic parsing approach.

3 Problem Definition & Approach

In this paper, we focus on using a knowledge base to answer *single-relation* questions. A single-relation question is defined as a question composed of an entity mention and a binary relation description, where the answer to this question would be an entity that has the relation with the given entity. An example of a single-relation question is “*When were DVD players invented?*” The entity is `dvd-player` and the relation is `be-invent-in`. The answer can thus be described as the following lambda expression:

$$\lambda x. \text{be-invent-in}(\text{dvd-player}, x)$$

$$Q \rightarrow RP \wedge M \quad (1)$$

$$RP \rightarrow \text{when were } X \text{ invented} \quad (2)$$

$$M \rightarrow \text{dvd players} \quad (3)$$

when were X invented

$$\rightarrow \text{be-invent-in} \quad (4)$$

dvd players

$$\rightarrow \text{dvd-player} \quad (5)$$

Figure 1: A potential semantic parse of the question “When were DVD players invented?”

A knowledge base in this work can be simply viewed as a collection of binary relation instances in the form of $r(e_1, e_2)$, where r is the relation and e_1 and e_2 are the first and second entity arguments.

Single-relation questions are perhaps the easiest form of questions that can directly be answered by a knowledge base. If the mapping of the relation and entity in the question can be correctly resolved, then the answer can be derived by a simple table lookup, assuming that the fact exists in the KB. However, due to the large number of paraphrases of the same question, identifying the mapping accurately remains a difficult problem.

Our approach in this work can be viewed as a simple semantic parser tailored to single-relation questions, powered by advanced semantic similarity models to handle the paraphrase issue. Given a question, we first separate it into two disjoint parts: the *entity mention* and the *relation pattern*. The entity mention is a subsequence of consecutive words in the question, where the relation pattern is the question where the mention is substituted by a special symbol. The mapping between the pattern and the relation in the KB, as well as the mapping between the mention and the entity are determined by corresponding semantic similarity models. The high-level approach can be viewed as a very simple context-free grammar, which is shown in Figure 1.

The probability of the rule in (1) is 1 since we assume the input is a single-relation question. For the exact decomposition of the question (e.g., (2), (3)), we simply enumerate all combinations and assign equal probabilities to them. The performance of this approach depends mainly on whether the relation pattern and entity mention can be resolved correctly (e.g., (4), (5)). To deter-

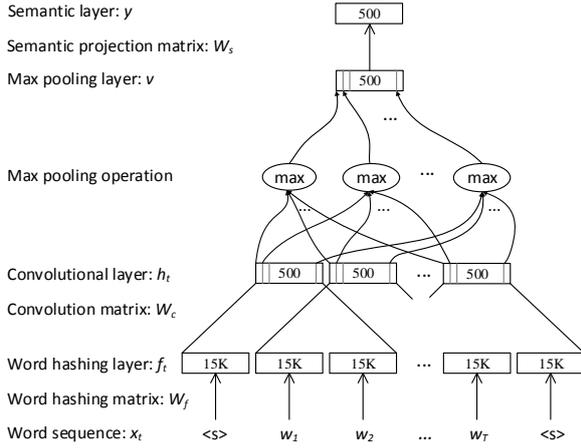


Figure 2: The CNNSM maps a variable-length word sequence to a low-dimensional vector in a latent semantic space. A word contextual window size (i.e., the receptive field) of three is used in the illustration. Convolution over word sequence via learned matrix W_c is performed implicitly via the earlier word hashing layer’s mapping with a local receptive field. The max operation across the sequence is applied for each of 500 feature dimensions separately.

mine the probabilities of such mappings, we propose using a semantic similarity model based on convolutional neural networks, which is the technical focus in this paper.

4 Convolutional Neural Network based Semantic Model

Following (Collobert et al., 2011; Shen et al., 2014b), we develop a new convolutional neural network (CNN) based semantic model (CNNSM) for semantic parsing. The CNNSM first uses a convolutional layer to project each word within a context window to a local contextual feature vector, so that semantically similar word- n -grams are projected to vectors that are close to each other in the contextual feature space. Further, since the overall meaning of a sentence is often determined by a few key words in the sentence, CNNSM uses a max pooling layer to extract the most salient local features to form a fixed-length global feature vector. The global feature vector can be then fed to feed-forward neural network layers to extract non-linear semantic features. The architecture of the CNNSM is illustrated in Figure 2. In what follows, we describe each layer of the CNNSM in detail, using the annotation illustrated in Figure 2.

In our model, we leverage the word hashing technique proposed in (Huang et al., 2013) where we first represent a word by a letter-trigram count vector. For example, given a word (e.g., cat), after adding word boundary symbols (e.g., #cat#), the word is segmented into a sequence of letter- n -grams (e.g., letter-trigrams: #c-a, c-a-t, a-t-#). Then, the word is represented as a count vector of letter-trigrams. For example, the letter-trigram representation of “cat” is:

$$f(\text{cat}) = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}$$

Indices of #c-a, c-a-t, a-t-# in the letter-tri-gram list, respectively.

In Figure 2, the word hashing matrix W_f denotes the transformation from a word to its letter-trigram count vector, which requires no learning. Word hashing not only makes the learning more scalable by controlling the size of the vocabulary, but also can effectively handle the OOV issues, sometimes due to spelling mistakes. Given the letter-trigram based word representation, we represent a word- n -gram by concatenating the letter-trigram vectors of each word, e.g., for the t -th word- n -gram at the word- n -gram layer, we have:

$$l_t = [f_{t-d}^T, \dots, f_t^T, \dots, f_{t+d}^T]^T, t = 1, \dots, T$$

where f_t is the letter-trigram representation of the t -th word, and $n = 2d + 1$ is the size of the contextual window. The convolution operation can be viewed as sliding window based feature extraction. It captures the word- n -gram contextual features. Consider the t -th word- n -gram, the convolution matrix projects its letter-trigram representation vector l_t to a contextual feature vector h_t . As shown in Figure 2, h_t is computed by

$$h_t = \tanh(W_c \cdot l_t), t = 1, \dots, T$$

where W_c is the feature transformation matrix, as known as the convolution matrix, which are shared among all word n -grams. The output of the convolutional layer is a sequence of local contextual feature vectors, one for each word (within a contextual window). Since many words do not have significant influence on the semantics of the sentence, we want to retain in the global feature vector only the salient features from a few key words. For this purpose, we use a max operation, also known as max pooling, to force the network to retain only

the most useful local features produced by the convolutional layers. Referring to the max-pooling layer of Figure 2, we have

$$v(i) = \max_{t=1, \dots, T} \{f_t(i)\}, i = 1, \dots, K$$

where $v(i)$ is the i -th element of the max pooling layer v , $h_t(i)$ is the i -th element of the t -th local feature vector h_t . K is the dimensionality of the max pooling layer, which is the same as the dimensionality of the local contextual feature vectors $\{h_t\}$. One more non-linear transformation layer is further applied on top of the global feature vector v to extract the high-level semantic representation, denoted by y . As shown in Figure 2, we have $y = \tanh(W_s \cdot v)$, where v is the global feature vector after max pooling, W_s is the semantic projection matrix, and y is the vector representation of the input query (or document) in latent semantic space. Given a pattern and a relation, we compute their relevance score by measuring the cosine similarity between their semantic vectors. The semantic relevance score between a pattern Q and a relation R is defined as the cosine score of their semantic vectors y_Q and y_R .

We train two CNN semantic models from sets of pattern–relation and mention–entity pairs, respectively. Following (Huang et al., 2013), for every pattern, the corresponding relation is treated as a positive example and 100 randomly selected other relations are used as negative examples. The setting for the mention–entity model is similar.

The posterior probability of the positive relation given the pattern is computed based on the cosine scores using softmax:

$$P(R^+|Q) = \frac{\exp(\gamma \cdot \cos(y_{R^+}, y_Q))}{\sum_{R'} \exp(\gamma \cdot \cos(y_{R'}, y_Q))}$$

where γ is a scaling factor set to 5. Model training is done by maximizing the log-posteriori using stochastic gradient descent. More detail can be found in (Shen et al., 2014a).

5 Experiments

In order to provide a fair comparison to previous work, we experimented with our approach using the PARALAX dataset (Fader et al., 2013), which consists of paraphrases of questions mined from WikiAnswers and answer triples from ReVerb. In this section, we briefly introduce the dataset, describe the system training and evaluation processes and, finally, present our experimental results.

5.1 Data & Model Training

The PARALEX training data consists of approximately 1.8 million pairs of questions and single-relation database queries, such as “When were DVD players invented?”, paired with `be-invent-in(dvd-player, ?)`. For evaluation, the authors further sampled 698 questions that belong to 37 clusters and hand labeled the answer triples returned by their systems.

To train our two CNN semantic models, we derived two parallel corpora based on the PARALEX training data. For relation patterns, we first scanned the original training corpus to see if there was an exact surface form match of the entity (e.g., `dvd-player` would map to “DVD player” in the question). If an exact match was found, then the pattern would be derived by replacing the mention in the question with the special symbol. The corresponding relation of this pattern was thus the relation used in the original database query, along with the variable argument position (i.e., 1 or 2, indicating whether the answer entity was the first or second argument of the relation). In the end, we derived about 1.2 million pairs of patterns and relations. We then applied these patterns to all the 1.8 million training questions, which helped discover 160 thousand new mentions that did not have the exact surface form matches to the entities.

When training the CNNSM for the pattern–relation similarity measure, we randomly split the 1.2 million pairs of patterns and relations into two sets: the training set of 1.19 million pairs, and the validation set of 12 thousand pairs for hyperparameter tuning. Data were tokenized by replacing hyphens with blank spaces. In the experiment, we used a context window (i.e., the receptive field) of three words in the convolutional neural networks. There were 15 thousand unique letter-trigrams observed in the training set (used for word hashing). Five hundred neurons were used in the convolutional layer, the max-pooling layer and the final semantic layer, respectively. We used a learning rate of 0.002 and the training converged after 150 iterations. A similar setting was used for the CNNSM for the mention–entity model, which was trained on 160 thousand mention–entity pairs.

5.2 Results

We used the same test questions in the PARALEX dataset to evaluate whether our system could find

	F ₁	Precision	Recall	MAP
CNNSM _{pm}	0.57	0.58	0.57	0.28
CNNSM _p	0.54	0.61	0.49	0.20
PARALEX	0.54	0.77	0.42	0.22

Table 1: Performance of two variations of our systems, compared with the PARALEX system.

the answers from the ReVerb database. Because our systems might find triples that were not returned by the PARALEX systems, we labeled these new question–triple pairs ourselves.

Given a question, the system first enumerated all possible decompositions of the mentions and patterns, as described earlier. We then computed the similarity scores between the pattern and all relations in the KB and retained 150 top-scoring relation candidates. For each selected relation, the system then checked all triples in the KB that had this relation and computed the similarity score between the mention and corresponding argument entity. The product of the probabilities of these two models, which are derived from the cosine similarity scores using softmax as described in Sec. 4, was used as the final score of the triple for ranking the answers. The top answer triple was used to compute the precision and recall of the system when reporting the system performance. By limiting the systems to output only answer triples with scores higher than a predefined threshold, we could control the trade-off between recall and precision and thus plot the precision–recall curve.

Table 1 shows the performance in F₁, precision, recall and mean average precision of our systems and PARALEX. We provide two variations here. CNNSM_{pm} is the full system and consists of two semantic similarity models for pattern–relation and mention–entity. The other model, CNNSM_p, only measures the similarity between the patterns and relations, and maps a mention to an entity when they have the same surface form.

Since the trade-off between precision and recall can be adjusted by varying the threshold, it is more informative to compare systems on the precision–recall curves, which are shown in Figure 3. As we can observe from the figure, the precision of our CNNSM_{pm} system is consistently higher than PARALEX across all recall regions. The CNNSM_m system also performs similarly to CNNSM_{pm} in the high precision regime, but is inferior when recall is higher. This is understandable

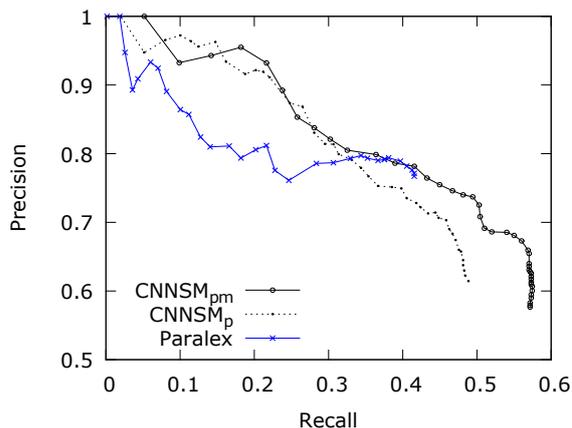


Figure 3: The precision–recall curves of the two variations of our systems and PARALEX.

since the system does not match mentions with entities of different surface forms (e.g., “Robert Hooke” to “Hooke”). Notice that the highest F₁ values of them are 0.61 and 0.56, compared to 0.54 of PARALEX. Tuning the thresholds using a validation set would be needed if there is a metric (e.g., F₁) that specifically needs to be optimized.

6 Conclusions

In this work, we propose a semantic parsing framework for single-relation questions. Compared to the existing work, our key insight is to match relation patterns and entity mentions using a semantic similarity function rather than lexical rules. Our similarity model is trained using convolutional neural networks with letter-trigrams vectors. This design helps the model go beyond bag-of-words representations and handles the OOV issue. Our method achieves higher precision on the QA task than the previous work, PARALEX, consistently at different recall points.

Despite the strong empirical performance, our system has room for improvement. For instance, due to the variety of entity mentions in the real world, the parallel corpus derived from the WikiAnswers data and ReVerb KB may not contain enough data to train a robust entity linking model. Replacing this component with a dedicated entity linking system could improve the performance and also reduce the number of pattern/mention candidates when processing each question. In the future, we would like to extend our method to other more structured KBs, such as Freebase, and to explore approaches to extend our system to handle multi-relation questions.

References

- Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA, October. Association for Computational Linguistics.
- David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.
- Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 423–433, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*.
- Scott Deerwester, Susan Dumais, Thomas Landauer, George Furnas, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6).
- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference of Empirical Methods in Natural Language Processing (EMNLP '11)*, Edinburgh, Scotland, UK, July 27–31.
- Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1608–1618, Sofia, Bulgaria, August. Association for Computational Linguistics.
- 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 *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2333–2338. ACM.
- Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1545–1556, Seattle, Washington, USA, October. Association for Computational Linguistics.
- Percy Liang, Michael I Jordan, and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics*, 39(2):389–446.
- Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. 2013. Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In *Interspeech*.
- Ruslan Salakhutdinov and Geoffrey Hinton. 2009. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978.
- Gerard Salton and Michael J. McGill. 1983. *Introduction to Modern Information Retrieval*. McGraw Hill.
- Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014a. A convolutional latent semantic model for web search. Technical Report MSR-TR-2014-55, Microsoft Research.
- Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014b. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion*, pages 373–374.
- Lappoon Tang and Raymond Mooney. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Machine Learning: ECML 2001*, pages 466–477. Springer.
- Gokhan Tur, Li Deng, Dilek Hakkani-Tur, and Xiaodong He. 2012. Towards deeper understanding: deep convex networks for semantic utterance classification. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 5045–5048. IEEE.
- John Zelle and Raymond Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1050–1055.

On WordNet Semantic Classes and Dependency Parsing

Kepa Bengoetxea[†], Eneko Agirre[†], Joakim Nivre[‡],
Yue Zhang^{*}, Koldo Gojenola[†]

[†]University of the Basque Country UPV/EHU / IXA NLP Group

[‡]Uppsala University / Department of Linguistics and Philology

^{*} Singapore University of Technology and Design

kepa.bengoetxea@ehu.es, e.agirre@ehu.es,

joakim.nivre@lingfil.uu.se, yue-zhang@sutd.edu.sg,

koldo.gojenola@ehu.es

Abstract

This paper presents experiments with WordNet semantic classes to improve dependency parsing. We study the effect of semantic classes in three dependency parsers, using two types of constituency-to-dependency conversions of the English Penn Treebank. Overall, we can say that the improvements are small and not significant using automatic POS tags, contrary to previously published results using gold POS tags (Agirre et al., 2011). In addition, we explore parser combinations, showing that the semantically enhanced parsers yield a small significant gain only on the more semantically oriented LTH treebank conversion.

1 Introduction

This work presents a set of experiments to investigate the use of lexical semantic information in dependency parsing of English. Whether semantics improve parsing is one interesting research topic both on parsing and lexical semantics. Broadly speaking, we can classify the methods to incorporate semantic information into parsers in two: systems using static lexical semantic repositories, such as WordNet or similar ontologies (Agirre et al., 2008; Agirre et al., 2011; Fujita et al., 2010), and systems using dynamic semantic clusters automatically acquired from corpora (Koo et al., 2008; Suzuki et al., 2009).

Our main objective will be to determine whether static semantic knowledge can help parsing. We will apply different types of semantic information to three dependency parsers. Specifically, we will test the following questions:

- Does semantic information in WordNet help dependency parsing? Agirre et al. (2011) found improvements in dependency parsing

using MaltParser on gold POS tags. In this work, we will investigate the effect of semantic information using predicted POS tags.

- Is the type of semantic information related to the type of parser? We will test three different parsers representative of successful paradigms in dependency parsing.
- How does the semantic information relate to the style of dependency annotation? Most experiments for English were evaluated on the Penn2Malt conversion of the constituency-based Penn Treebank. We will also examine the LTH conversion, with richer structure and an extended set of dependency labels.
- How does WordNet compare to automatically obtained information? For the sake of comparison, we will also perform the experiments using syntactic/semantic clusters automatically acquired from corpora.
- Does parser combination benefit from semantic information? Different parsers can use semantic information in diverse ways. For example, while MaltParser can use the semantic information in local contexts, MST can incorporate them in global contexts. We will run parser combination experiments with and without semantic information, to determine whether it is useful in the combined parsers.

After introducing related work in section 2, section 3 describes the treebank conversions, parsers and semantic features. Section 4 presents the results and section 5 draws the main conclusions.

2 Related work

Broadly speaking, we can classify the attempts to add external knowledge to a parser in two sets: using large semantic repositories such as WordNet and approaches that use information automatically acquired from corpora. In the first group, Agirre et al. (2008) trained two state-of-the-art constituency-based statistical parsers (Charniak,

2000; Bikel, 2004) on semantically-enriched input, substituting content words with their semantic classes, trying to overcome the limitations of lexicalized approaches to parsing (Collins, 2003) where related words, like *scissors* and *knife*, cannot be generalized. The results showed a significant improvement, giving the first results over both WordNet and the Penn Treebank (PTB) to show that semantics helps parsing. Later, Agirre et al. (2011) successfully introduced WordNet classes in a dependency parser, obtaining improvements on the full PTB using gold POS tags, trying different combinations of semantic classes. MacKinlay et al. (2012) investigate the addition of semantic annotations in the form of word sense hypernyms, in HPSG parse ranking, reducing error rate in dependency F-score by 1%, while some methods produce substantial decreases in performance. Fujita et al. (2010) showed that fully disambiguated sense-based features smoothed using ontological information are effective for parse selection.

On the second group, Koo et al. (2008) presented a semisupervised method for training dependency parsers, introducing features that incorporate word clusters automatically acquired from a large unannotated corpus. The clusters include strongly semantic associations like {apple, pear} or {Apple, IBM} and also syntactic clusters like {of, in}. They demonstrated its effectiveness in dependency parsing experiments on the PTB and the Prague Dependency Treebank. Suzuki et al. (2009), Sagae and Gordon (2009) and Candito and Seddah (2010) also experiment with the same cluster method. Recently, Täckström et al. (2012) tested the incorporation of cluster features from unlabeled corpora in a multilingual setting, giving an algorithm for inducing cross-lingual clusters.

3 Experimental Framework

In this section we will briefly describe the PTB-based datasets (subsection 3.1), followed by the data-driven parsers used for the experiments (subsection 3.2). Finally, we will describe the different types of semantic representation that were used.

3.1 Treebank conversions

*Penn2Malt*¹ performs a simple and direct conversion from the constituency-based PTB to a dependency treebank. It obtains projective trees and has been used in several works, which allows us to

¹<http://w3.msi.vxu.se/nivre/research/Penn2Malt.html>

compare our results with related experiments (Koo et al., 2008; Suzuki et al., 2009; Koo and Collins, 2010). We extracted dependencies using standard head rules (Yamada and Matsumoto, 2003), and a reduced set of 12 general dependency tags.

*LTH*² (Johansson and Nugues, 2007) presents a conversion better suited for semantic processing, with a richer structure and a more fine-grained set of dependency labels (42 different dependency labels), including links to handle long-distance phenomena, giving a 6.17% of nonprojective sentences. The results from parsing the LTH output are lower than those for Penn2Malt conversions.

3.2 Parsers

We have made use of three parsers representative of successful paradigms in dependency parsing.

MaltParser (Nivre et al., 2007) is a deterministic transition-based dependency parser that obtains a dependency tree in linear-time in a single pass over the input using a stack of partially analyzed items and the remaining input sequence, by means of history-based feature models. We added two features that inspect the semantic feature at the top of the stack and the next input token.

*MST*³ represents global, exhaustive graph-based parsing (McDonald et al., 2005; McDonald et al., 2006) that finds the highest scoring directed spanning tree in a graph. The learning procedure is global since model parameters are set relative to classifying the entire dependency graph, in contrast to the local but richer contexts used by transition-based parsers. The system can be trained using first or second order models. The second order projective algorithm performed best on both conversions, and we used it in the rest of the evaluations. We modified the system in order to add semantic features, combining them with wordforms and POS tags, on the parent and child nodes of each arc.

*ZPar*⁴ (Zhang and Clark, 2008; Zhang and Nivre, 2011) performs transition-based dependency parsing with a stack of partial analysis and a queue of remaining inputs. In contrast to *MaltParser* (local model and greedy deterministic search) *ZPar* applies global discriminative learning and beam search. We extend the feature set of *ZPar* to include semantic features. Each set of semantic information is represented by two atomic

²http://nlp.cs.lth.se/software/treebank_converter

³<http://mstpaser.sourceforge.net>

⁴www.sourceforge.net/projects/zpar

	Base line	WordNet SF	WordNet SS	Clusters
Malt	88.46	88.49 (+0.03)	88.42 (-0.04)	88.59 (+0.13)
MST	90.55	90.70 (+0.15)	90.47 (-0.08)	90.88 (+0.33)‡
ZPar	91.52	91.65 (+0.13)	91.70 (+0.18)†	91.74 (+0.22)

Table 1: LAS results with several parsing algorithms, Penn2Malt conversion (†: $p < 0.05$, ‡: $p < 0.005$). In parenthesis, difference with baseline.

feature templates, associated with the top of the stack and the head of the queue, respectively. ZPar was directly trained on the Penn2Malt conversion, while we applied the pseudo-projective transformation (Nilsson et al., 2008) on LTH, in order to deal with non-projective arcs.

3.3 Semantic information

Our aim was to experiment with different types of WordNet-related semantic information. For comparison with automatically acquired information, we will also experiment with bit clusters.

WordNet. We will experiment with the semantic representations used in Agirre et al. (2008) and Agirre et al. (2011), based on WordNet 2.1. WordNet is organized into sets of synonyms, called synsets (SS). Each synset in turn belongs to a unique semantic file (SF). There are a total of 45 SFs (1 for adverbs, 3 for adjectives, 15 for verbs, and 26 for nouns), based on syntactic and semantic categories. For example, noun SFs differentiate nouns denoting acts or actions, and nouns denoting animals, among others. We experiment with both full SSs and SFs as instances of fine-grained and coarse-grained semantic representation, respectively. As an example, *knife* in its tool sense is in the EDGE TOOL USED AS A CUTTING INSTRUMENT singleton synset, and also in the ARTIFACT SF along with thousands of words including *cutter*. These are the two extremes of semantic granularity in WordNet. For each semantic representation, we need to determine the semantics of each occurrence of a target word. Agirre et al. (2011) used i) gold-standard annotations from SemCor, a subset of the PTB, to give an upper bound performance of the semantic representation, ii) first sense, where all instances of a word were tagged with their most frequent sense, and iii) automatic sense ranking, predicting the most frequent sense for each word (McCarthy et al., 2004). As we will make use of the full PTB, we only have access to the first sense information.

Clusters. Koo et al. (2008) describe a semi-

	Base line	WordNet SF	WordNet SS	Clusters
Malt	84.95	85.12 (+0.17)	85.08 (+0.16)	85.13 (+0.18)
MST	85.06	85.35 (+0.29)‡	84.99 (-0.07)	86.18 (+1.12)‡
ZPar	89.15	89.33 (+0.18)	89.19 (+0.04)	89.17 (+0.02)

Table 2: LAS results with several parsing algorithms in the LTH conversion (†: $p < 0.05$, ‡: $p < 0.005$). In parenthesis, difference with baseline.

supervised approach that makes use of cluster features induced from unlabeled data, providing significant performance improvements for supervised dependency parsers on the Penn Treebank for English and the Prague Dependency Treebank for Czech. The process defines a hierarchical clustering of the words, which can be represented as a binary tree where each node is associated to a bit-string, from the more general (root of the tree) to the more specific (leaves). Using prefixes of various lengths, it can produce clusterings of different granularities. It can be seen as a representation of syntactic-semantic information acquired from corpora. They use short strings of 4-6 bits to represent parts of speech and the full strings for wordforms.

4 Results

In all the experiments we employed a baseline feature set using word forms and parts of speech, and an enriched feature set (WordNet or clusters). We firstly tested the addition of each individual semantic feature to each parser, evaluating its contribution to the parser’s performance. For the combinations, instead of feature-engineering each parser with the wide array of different possibilities for features, as in Agirre et al. (2011), we adopted the simpler approach of combining the outputs of the individual parsers by voting (Sagae and Lavie, 2006). We will use Labeled Attachment Score (LAS) as our main evaluation criteria. As in previous work, we exclude punctuation marks. For all the tests, we used a perceptron POS-tagger (Collins, 2002), trained on WSJ sections 2–21, to assign POS tags automatically to both the training (using 10-way jackknifing) and test data, obtaining a POS tagging accuracy of 97.32% on the test data. We will make use of Bikel’s randomized parsing evaluation comparator to test the statistical significance of the results. In all of the experiments the parsers were trained on sections 2-21 of the PTB and evaluated on the development set (section 22). Finally, the best performing system was evaluated on the test set (section 23).

Parsers	LAS	UAS
Best baseline (ZPar)	91.52	92.57
Best single parser (ZPar + Clusters)	91.74 (+0.22)	92.63
Best combination (3 baseline parsers)	91.90 (+0.38)	93.01
Best combination of 3 parsers: 3 baselines + 3 SF extensions	91.93 (+0.41)	92.95
Best combination of 3 parsers: 3 baselines + 3 SS extensions	91.87 (+0.35)	92.92
Best combination of 3 parsers: 3 baselines + 3 cluster extensions	91.90 (+0.38)	92.90

Table 3: Parser combinations on Penn2Malt.

Parsers	LAS	UAS
Best baseline (ZPar)	89.15	91.81
Best single parser (ZPar + SF)	89.33 (+0.15)	92.01
Best combination (3 baseline parsers)	89.15 (+0.00)	91.81
Best combination of 3 parsers: 3 baselines + 3 SF extensions	89.56 (+0.41) [‡]	92.23
Best combination of 3 parsers: 3 baselines + 3 SS extensions	89.43 (+0.28)	93.12
Best combination of 3 parsers: 3 baselines + 3 cluster extensions	89.52 (+0.37) [†]	92.19

Table 4: Parser combinations on LTH ([†]: $p < 0.05$, [‡]: $p < 0.005$).

4.1 Single Parsers

We run a series of experiments testing each individual semantic feature, also trying different learning configurations for each one. Regarding the WordNet information, there were 2 different features to experiment with (SF and SS). For the bit clusters, there are different possibilities, depending on the number of bits used. For Malt and MST, all the different lengths of bit strings were used. Given the computational requirements and the previous results on Malt and MST, we only tested all bits in ZPar. Tables 1 and 2 show the results.

Penn2Malt. Table 1 shows that the only significant increase over the baseline is for ZPar with SS and for MST with clusters.

LTH. Looking at table 2, we can say that the differences in baseline parser performance are accentuated when using the LTH treebank conversion, as ZPar clearly outperforms the other two parsers by more than 4 absolute points. We can see that SF helps all parsers, although it is only significant for MST. Bit clusters improve significantly MST, with the highest increase across the table.

Overall, we see that the small improvements do not confirm the previous results on Penn2Malt, MaltParser and gold POS tags. We can also conclude that automatically acquired clusters are specially effective with the MST parser in both treebank conversions, which suggests that the type of semantic information has a direct relation to the parsing algorithm. Section 4.3 will look at the details by each knowledge type.

4.2 Combinations

Subsection 4.1 presented the results of the base algorithms and their extensions based on semantic features. Sagae and Lavie (2006) report improvements over the best single parser when combining three transition-based models and one graph-based model. The same technique was also used by the winning team of the CoNLL 2007 Shared Task (Hall et al., 2007), combining six transition-based parsers. We used MaltBlender⁵, a tool for merging the output of several dependency parsers, using the Chu-Liu/Edmonds directed MST algorithm. After several tests we noticed that weighted voting by each parser’s labeled accuracy gave good results, using it in the rest of the experiments. We trained different types of combination:

- Base algorithms. This set includes the 3 baseline algorithms, MaltParser, MST, and ZPar.
- Extended parsers, adding semantic information to the baselines. We include the three base algorithms and their semantic extensions (SF, SS, and clusters). It is known (Surdeanu and Manning, 2010) that adding more parsers to an ensemble usually improves accuracy, as long as they add to the diversity (and almost regardless of their accuracy level). So, for the comparison to be fair, we will compare ensembles of 3 parsers, taken from sets of 6 parsers (3 baselines + 3 SF, SS, and cluster extensions, respectively).

In each experiment, we took the best combination of individual parsers on the development set for the final test. Tables 3 and 4 show the results.

Penn2Malt. Table 3 shows that the combination of the baselines, without any semantic information, considerably improves the best baseline. Adding semantics does not give a noticeable increase with respect to combining the baselines.

LTH (table 4). Combining the 3 baselines does not give an improvement over the best baseline, as ZPar clearly outperforms the other parsers. However, adding the semantic parsers gives an increase with respect to the best single parser (ZPar + SF), which is small but significant for SF and clusters.

4.3 Analysis

In this section we analyze the data trying to understand where and how semantic information helps most. One of the obstacles of automatic parsers is the presence of incorrect POS tags due to auto-

⁵<http://w3.msi.vxu.se/users/jni/blend/>

POS tags	Parser	LAS test set	LAS on sentences without POS errors	LAS on sentences with POS errors
Gold	ZPar	90.45	91.68	89.14
Automatic	ZPar	89.15	91.62	86.51
Automatic	Best combination of 3 parsers: 3 baselines + 3 SF extensions	89.56 (+0.41)	91.90 (+0.28)	87.06 (+0.55)
Automatic	Best combination of 3 parsers: 3 baselines + 3 SS extensions	89.43 (+0.28)	91.95 (+0.33)	86.75 (+0.24)
Automatic	Best combination of 3 parsers: 3 baselines + 3 cluster extensions	89.52 (+0.37)	91.92 (+0.30)	86.96 (+0.45)

Table 5: Differences in LAS (LTH) for baseline and extended parsers with sentences having correct/incorrect POS tags (the parentheses show the difference w.r.t ZPar with automatic POS tags).

matic tagging. For example, ZPar’s LAS score on the LTH conversion drops from 90.45% with gold POS tags to 89.12% with automatic POS tags. We will examine the influence of each type of semantic information on sentences that contain or not POS errors, and this will clarify whether the increments obtained when using semantic information are useful for correcting the negative influence of POS errors or they are orthogonal and constitute a source of new information independent of POS tags. With this objective in mind, we analyzed the performance on the subset of the test corpus containing the sentences which had POS errors (1,025 sentences and 27,300 tokens) and the subset where the sentences had (automatically assigned) correct POS tags (1,391 sentences and 29,386 tokens).

Table 5 presents the results of the best single parser on the LTH conversion (ZPar) with gold and automatic POS tags in the first two rows. The LAS scores are particularized for sentences that contain or not POS errors. The following three rows present the enhanced (combined) parsers that make use of semantic information. As the combination of the three baseline parsers did not give any improvement over the best single parser (ZPar), we can hypothesize that the gain coming from the parser combinations comes mostly from the addition of semantic information. Table 5 suggests that the improvements coming from WordNet’s semantic file (SF) are unevenly distributed between the sentences that contain POS errors and those that do not (an increase of 0.28 for sentences without POS errors and 0.55 for those with errors). This could mean that a big part of the information contained in SF helps to alleviate the errors performed by the automatic POS tagger. On the other hand, the increments are more evenly distributed for SS and clusters, and this can be due to the fact that the semantic information is orthogonal to the POS, giving similar improvements for sentences that contain or not POS errors.

We independently tested this fact for the individual parsers. For example, with MST and SF the gains almost doubled for sentences with incorrect POS tags (+0.37 with respect to +0.21 for sentences with correct POS tags) while the gains of adding clusters’ information for sentences without and with POS errors were similar (0.91 and 1.33, respectively). This aspect deserves further investigation, as the improvements seem to be related to both the type of semantic information and the parsing algorithm. We did an initial exploration but it did not give any clear indication of the types of improvements that could be expected using each parser and semantic data.

5 Conclusions

This work has tried to shed light on the contribution of semantic information to dependency parsing. The experiments were thorough, testing two treebank conversions and three parsing paradigms on automatically predicted POS tags. Compared to (Agirre et al., 2011), which used MaltParser on the LTH conversion and gold POS tags, our results can be seen as a negative outcome, as the improvements are very small and non-significant in most of the cases. For parser combination, WordNet semantic file information does give a small significant increment in the more fine-grained LTH representation. In addition we show that the improvement of automatic clusters is also weak. For the future, we think different parsers, either a more elaborate scheme is needed for word classes, requiring to explore different levels of generalization in the WordNet (or alternative) hierarchies.

Acknowledgments

This research was supported by the the Basque Government (IT344- 10, S PE11UN114), the University of the Basque Country (GIU09/19) and the Spanish Ministry of Science and Innovation (MICINN, TIN2010-20218).

References

- Eneko Agirre, Timothy Baldwin, and David Martinez. 2008. Improving parsing and PP attachment performance with sense information. In *Proceedings of ACL-08: HLT*, pages 317–325, Columbus, Ohio, June. Association for Computational Linguistics.
- Eneko Agirre, Kepa Bengoetxea, Koldo Gojenola, and Joakim Nivre. 2011. Improving dependency parsing with semantic classes. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 699–703, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Daniel M. Bikel. 2004. Intricacies of collins’ parsing model. *Computational Linguistics*, 30(4):479–511.
- Marie Candito and Djamel Seddah. 2010. Parsing word clusters. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 76–84, Los Angeles, CA, USA, June. Association for Computational Linguistics.
- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, NAACL 2000, pages 132–139, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 1–8. Association for Computational Linguistics, July.
- Michael Collins. 2003. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637, December.
- Sanae Fujita, Francis Bond, Stephan Oepen, and Takaaki Tanaka. 2010. Exploiting semantic information for hpsg parse selection. *Research on Language and Computation*, 8(1):122.
- Johan Hall, Jens Nilsson, Joakim Nivre, Glsen Eryigit, Beta Megyesi, Mattias Nilsson, and Markus Saers. 2007. Single malt or blended? a study in multilingual parser optimization. In *Proceedings of the CoNLL Shared Task EMNLP-CoNLL*.
- Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In *Proceedings of NODALIDA 2007*, pages 105–112, Tartu, Estonia, May 25–26.
- 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, July. Association for Computational Linguistics.
- Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, Ohio, June. Association for Computational Linguistics.
- Andrew MacKinlay, Rebecca Dridan, Diana McCarthy, and Timothy Baldwin. 2012. The effects of semantic annotations on precision parse ranking. In *First Joint Conference on Lexical and Computational Semantics (*SEM)*, page 228236, Montreal, Canada, June. Association for Computational Linguistics.
- Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2004. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL’04), Main Volume*, pages 279–286, Barcelona, Spain, July.
- R. McDonald, K. Crammer, and F. Pereira. 2005. On-line large-margin training of dependency parsers. In *Proceedings of ACL*.
- R. McDonald, K. Lerman, and F. Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of CoNLL 2006*.
- Jens Nilsson, Joakim Nivre, and Johan Hall. 2008. Generalizing tree transformations for inductive dependency parsing. In *Proceedings of the 45th Conference of the ACL*.
- Joakim Nivre, Johan Hall, Jens Nilsson, Chaney A., Glsen Eryit, Sandra Kbler, Marinov S., and Edwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*.
- Kenji Sagae and Andrew Gordon. 2009. Clustering words by syntactic similarity improves dependency parsing of predicate-argument structures. In *Proceedings of the Eleventh International Conference on Parsing Technologies*.
- Kenji Sagae and Alon Lavie. 2006. Parser combination by reparsing. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*.
- Mihai Surdeanu and Christopher D. Manning. 2010. Ensemble models for dependency parsing: Cheap and good? In *Proceedings of the North American Chapter of the Association for Computational Linguistics Conference (NAACL-2010)*, Los Angeles, CA, June.
- Jun Suzuki, Hideki Isozaki, Xavier Carreras, and Michael Collins. 2009. An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 551–560, Singapore, August. Association for Computational Linguistics.

- Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 477–487, Montréal, Canada, June. Association for Computational Linguistics.
- Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th IWPT*, pages 195–206. Association for Computational Linguistics.
- Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193, Portland, Oregon, USA, June. Association for Computational Linguistics.

Enforcing Structural Diversity in Cube-pruned Dependency Parsing

Hao Zhang Ryan McDonald

Google, Inc.

{haozhang, ryanmcd}@google.com

Abstract

In this paper we extend the cube-pruned dependency parsing framework of Zhang et al. (2012; 2013) by forcing inference to maintain both label and structural ambiguity. The resulting parser achieves state-of-the-art accuracies, in particular on datasets with a large set of dependency labels.

1 Introduction

Dependency parsers assign a syntactic dependency tree to an input sentence (Kübler et al., 2009), as exemplified in Figure 1. Graph-based dependency parsers parameterize models directly over substructures of the tree, including single arcs (McDonald et al., 2005), sibling or grandparent arcs (McDonald and Pereira, 2006; Carreras, 2007) or higher-order substructures (Koo and Collins, 2010; Ma and Zhao, 2012). As the scope of each feature function increases so does parsing complexity, e.g., $o(n^5)$ for fourth-order dependency parsing (Ma and Zhao, 2012). This has led to work on approximate inference, typically via pruning (Bergsma and Cherry, 2010; Rush and Petrov, 2012; He et al., 2013)

Recently, it has been shown that cube-pruning (Chiang, 2007) can efficiently introduce higher-order dependencies in graph-based parsing (Zhang and McDonald, 2012). Cube-pruned dependency parsing runs standard bottom-up chart parsing using the lower-order algorithms. Similar to k -best inference, each chart cell maintains a beam of k -best partial dependency structures. Higher-order features are scored when combining beams during inference. Cube-pruning is an approximation, as the highest scoring tree may fall out of the beam before being fully scored with higher-order features. However, Zhang et al. (2013) observe state-of-the-art results when training accounts for errors that arise due to such approximations.

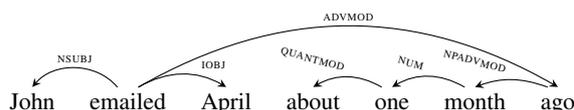


Figure 1: A sample dependency parse.

In this work we extend the cube-pruning framework of Zhang et al. by observing that dependency parsing has two fundamental sources of ambiguity. The first, structural ambiguity, pertains to confusions about the unlabeled structure of the tree, e.g., the classic prepositional phrase attachment problem. The second, label ambiguity, pertains to simple label confusions, e.g., whether a verbal object is direct or indirect.

Distinctions between arc labels are frequently fine-grained and easily confused by parsing models. For example, in the Stanford dependency label set (De Marneffe et al., 2006), the labels TMOD (temporal modifier), NPADVMOD (noun-phrase adverbial modifier), IOBJ (indirect object) and DOBJ (direct object) can all be noun phrases that modify verbs to their right. In the context of cube-pruning, during inference, the system opts to maintain a large amount of label ambiguity at the expense of structural ambiguity. Frequently, the beam stores only label ambiguities and the resulting set of trees have identical unlabeled structure. For example, in Figure 1, the aforementioned label ambiguity around noun objects to the right of the verb (DOBJ vs. IOBJ vs. TMP) could lead one or more of the structural ambiguities falling out of the beam, especially if the beam is small.

To combat this, we introduce a secondary beam for each unique unlabeled structure. That is, we partition the primary (entire) beam into disjoint groups according to the identity of unlabeled structure. By limiting the size of the secondary beam, we restrict label ambiguity and enforce structural diversity within the primary beam. The resulting parser consistently improves on the state-of-the-art parser of Zhang et al. (2013). In

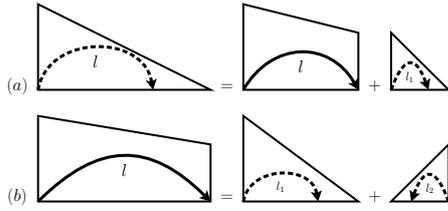


Figure 2: Structures and rules for parsing with the (Eisner, 1996) algorithm. Solid lines show only the construction of right-pointing first-order dependencies. l is the predicted arc label. Dashed lines are the additional sibling modifier signatures in a generalized algorithm, specifically the previous modifier in complete chart items.

particular, data sets with large label sets (and thus a large number of label confusions) typically see the largest jumps in accuracy. Finally, we show that the same result cannot be achieved by simply increasing the size of the beam, but requires explicit enforcing of beam diversity.

2 Structural Diversity in Cube-Pruning

Our starting point is the cube-pruned dependency parsing model of Zhang and McDonald (2012). In that work, as here, inference is simply the Eisner first-order parsing model (Eisner, 1996) shown in Figure 2. In order to score higher-order features, each chart item maintains a list of signatures, which represent subtrees consistent with the chart item. The stored signatures are the relevant portions of the subtrees that will be part of higher-order feature calculations. For example, to score features over adjacent arcs, we might maintain additional signatures, again shown in Figure 2.

The scope of the signature adds asymptotic complexity to parsing. Even for second-order siblings, there will now be $O(n)$ possible signatures per chart item. The result is that parsing complexity increases from $O(n^3)$ to $O(n^5)$. Instead of storing all signatures, Zhang and McDonald (2012) store the current k -best in a beam. This results in approximate inference, as some signatures may fall out of the beam before higher-order features can be scored. This general trick is known as *cube-pruning* and is a common approach to dealing with large hypergraph search spaces in machine translation (Chiang, 2007).

Cube-pruned parsing is analogous to k -best parsing algorithmically. But there is a fundamental difference. In k -best parsing, if two subtrees t_a and t_b belong to the same chart item, with t_a

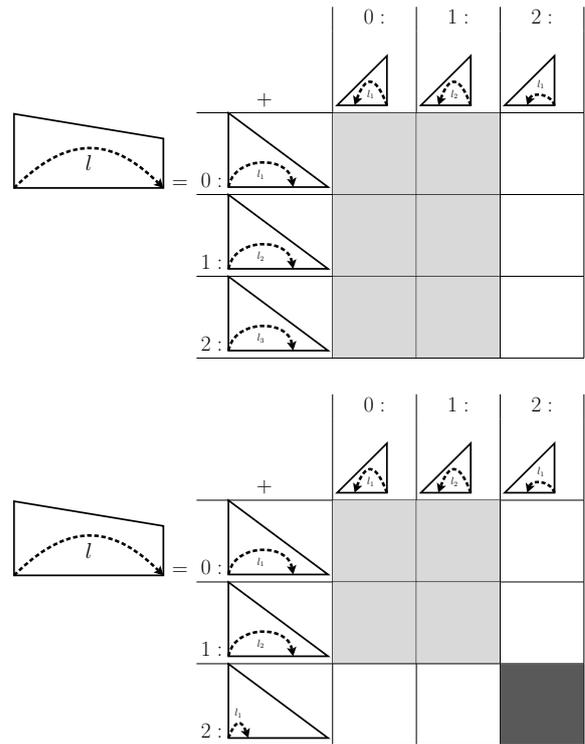


Figure 3: Merging procedure in cube pruning. The bottom shows that enforcing diversity in the k -best lists can give chance to a good structure at (2, 2).

ranking higher than t_b , then an extension of t_a through combing with a subtree t_c from another chart item must also score higher than that of t_b . This property is called the *monotonicity property*. Based on it, k -best parsing merges k -best subtrees in the following way: given two chart items with k -best lists to be combined, it proceeds on the two sorted lists monotonically from beginning to end to generate combinations. Cube pruning follows the merging procedure despite the loss of monotonicity due to the addition of higher-order feature functions over the signatures of the subtrees. The underlying assumption of cube pruning is that the true k -best results are likely in the cross-product space of top-ranked component subtrees. Figure 3 shows that the space is the top-left corner of the grid in the binary branching cases.

As mentioned earlier, the elements in chart item k -best lists are feature signatures of subtrees. We make a distinction between *labeled signatures* and *unlabeled signatures*. As feature functions are defined on sub-graphs of the dependency trees, a feature signature is labeled if and only if feature functions draw information from both the arcs in the sub-graph and the labels on the arcs. Every labeled signature projects to an unlabeled signature

that ignores the arc labels.

The motivation for introducing unlabeled signatures for labeled parsing is to enforce structural diversity. Figure 3 illustrates the idea. In the top diagram, there is only one unlabeled signature in one of the two lists. This is likely to happen when there is label ambiguity so that all three labels have similar scores. In such cases, alternative tree structures further down in the list that have the potential to be scored higher when incorporating higher-order features, lose this opportunity due to pruning. By contrast, if we introduce structural diversity by limiting the number of label variants, such alternative structures can come out on top.

More formally, when the feature signatures of the subtrees include arc labels, the cardinality of the set of all possible signatures grows by a polynomial of the size of the label set. This factor has a diluting effect on the diversity of unlabeled signatures within the beam. The larger the label set is, the greater the chance label ambiguity will dominate the beam. Therefore, we introduce a second level of beam specifically for labeled signatures. We call it the *secondary beam*, relative to the *primary beam*, i.e., the entire beam. The secondary beam limits the number of labeled signatures for each unlabeled signature, a projection of labeled signature, while the primary beam limits the total number of labeled signatures. To illustrate this, consider an original primary beam of length b and a secondary beam length of s_b . Let t_i^j represent the i^{th} highest scoring labeled variant of unlabeled structure j . The table below shows a specific example of beam configurations for $b = 4$ for all possible values of s_b . The original beam is the pathological case where all signatures have the same unlabeled projection. When $s_b = 1$, all signatures in the beam now have a different unlabeled projection. When $s_b = 4$, the beam reverts to the original without any structural diversity. Values between balance structural and label diversity.

beam rank	original $b=4$	$b = 4$ $s_b = 1$	$b = 4$ $s_b = 2$	$b = 4$ $s_b = 3$	$b = 4$ $s_b = 4$
1	t_1^1	t_1^1	t_1^1	t_1^1	t_1^1
2	t_2^1	t_2^2	t_2^2	t_2^2	t_2^2
3	t_3^1	t_3^3	t_3^2	t_3^3	t_3^3
4	t_4^1	t_4^4	t_4^3	t_4^2	t_4^4
..... beam cut-off					
5	t_1^2
6	t_1^3
7	t_2^2
8	t_3^2
9	t_4^4

To achieve this in cube pruning, deeper exploration in the merging procedure becomes necessary. In this example, originally the merging procedure stops when t_4^1 has been explored. When $s_b = 1$, the exploration needs to go further from rank 4 to 9. When $s_b = 2$, it needs to go from 4 to 6. When $s_b = 3$, only one more step to rank 5 is necessary. The amount of additional computation depends on the value of s_b , the composition of the incoming k -best lists, and the feature functions which determine feature signatures. To account for this we also compare to baselines systems that simply increase the size of the beam to a comparable run-time.

In our experiments we found that $s_b = b/2$ is typically a good choice. As in most parsing systems, beams are applied consistently during learning and testing because feature weights will be adjusted according to the diversity of the beam.

3 Experiments

We use the cube-pruned dependency parser of Zhang et al. (2013) as our baseline system. To make an apples-to-apples comparison, we use the same online learning algorithm and the same feature templates. The feature templates include first-to-third-order labeled features and valency features. More details of these features are described in Zhang and McDonald (2012). For online learning, we apply the same violation-fixing strategy (so-called single-node max-violation) on MIRA and run 8 epochs of training for all experiments.

For English, we conduct experiments on the commonly-used constituency-to-dependency-converted Penn Treebank data sets. The first one, Penn-YM, was created by the Penn2Malt¹ software. The second one, Penn-S-2.0.5, used the Stanford dependency framework (De Marneffe et al., 2006) by applying version 2.0.5 of the Stanford parser. The third one, Penn-S-3.3.0 was converted by version 3.3.0 of the Stanford parser. The train/dev/test split was standard: sections 2-21 for training; 22 for validation; and 23 for evaluation. Automatic POS tags for Penn-YM and Penn-S-2.0.5 are provided by TurboTagger (Martins et al., 2013) with an accuracy of 97.3% on section 23. For Chinese, we use the CTB-5 dependency treebank which was converted from the original constituent treebank by Zhang and Nivre (2011) and use gold-standard POS tags as is standard.

¹<http://stp.lingfil.uu.se/~nivre/research/Penn2Malt.html>

	<i>Berkeley Parser</i>		<i>TurboParser</i>		<i>Cube-pruned w/o diversity</i>		<i>Cube-pruned w/ diversity</i>	
	<i>UAS</i>	<i>LAS</i>	<i>UAS</i>	<i>LAS</i>	<i>UAS</i>	<i>LAS</i>	<i>UAS</i>	<i>LAS</i>
PENN-YM	-	-	93.07	-	93.50	92.41	93.57	92.48
PENN-S-2.0.5	-	-	92.82	-	93.59	91.17	93.71	91.37
PENN-S-3.3.0	93.31	91.01	92.20	89.67	92.91	90.52	93.01	90.64
PENN-S-3.3.0-GOLD	93.65	92.05	93.56	91.99	94.32	92.90	94.40	93.02
CTB-5	-	-	-	-	87.78	86.13	87.96	86.34

Table 1: English and Chinese results for cube pruning dependency parsing with the enforcement of structural diversity. PENN-S and CTB-5 are significant at $p < 0.05$. Penn-S-2.0.5 TurboParser result is from Martins et al. (2013). Following Kong and Smith (2014), we trained our models on Penn-S-3.3.0 with gold POS tags and evaluated with both non-gold (Stanford tagger) and gold tags.

Table 1 shows the main results of the paper. Both the baseline and the new system keep a beam of size 6 for each chart cell. The difference is that the new system enforces structural diversity with the introduction of a secondary beam for label variants. We choose the secondary beam that yields the highest LAS on the development data sets for Penn-YM, Penn-S-2.0.5 and CTB-5. Indeed we observe larger improvements for the data sets with larger label sets. Penn-S-2.0.5 has 49 labels and observes a 0.2% absolute improvement in LAS. Although CTB-5 has a small label set (18), we do see similar improvements for both UAS and LAS. There is a slight improvement for Penn-YM despite the fact that Penn-YM has the most compact label set (12). These results are the highest known in the literature. For the Penn-S-3.3.0 results we can see that our model outperforms TurboParser and is competitive with the Berkeley constituency parser (Petrov et al., 2006). In particular, if gold tags are assumed, cube-pruning significantly outperforms Berkeley. This suggests that joint tagging and parsing should improve performance further in the non-gold tag setting, as that is a differentiating characteristic of constituency parsers. Table 2 shows the results on the CoNLL 2006/2007 data sets (Buchholz and Marsi, 2006; Nivre et al., 2007). For simplicity, we set the secondary beam to 3 for all. We can see that overall there is an improvement in accuracy and this is highly correlated with the size of the label set.

In order to examine the importance of balancing structural diversity and labeled diversity, we let the size of the secondary beam vary from one to the size of the primary beam. In Table 3, we show the results of all combinations of beam settings of primary beam sizes 4 and 6 for three data sets: Penn-YM, Penn-S-2.0.5, and CTB-5 respectively. In the table, we highlight the best results for each beam size and data set on the development data. For 5 of the total of 6 comparison groups – three lan-

<i>Language(labels)</i>	<i>w/o diversity</i>		<i>w/ diversity</i>	
	<i>UAS</i>	<i>LAS</i>	<i>UAS</i>	<i>LAS</i>
CZECH(82)	88.36	82.16	88.36	82.02
SWEDISH(64)	91.62	85.08	91.85	85.26
PORTUGUESE(55)	92.07	88.30	92.23	88.50
DANISH(53)	91.88	86.95	91.78	86.93
HUNGARIAN(49)	85.85	81.02	86.55	81.79
GREEK(46)	86.14	78.20	86.21	78.45
GERMAN(46)	92.03	89.44	92.01	89.52
CATALAN(42)	94.58	89.05	94.91	89.54
BASQUE(35)	79.59	71.52	80.14	71.94
ARABIC(27)	80.48	69.68	80.56	69.98
TURKISH(26)	76.94	66.80	77.14	67.00
SLOVENE(26)	86.01	77.14	86.27	77.44
DUTCH(26)	83.57	80.29	83.39	80.19
ITALIAN(22)	87.57	83.22	87.38	82.95
SPANISH(21)	87.96	84.95	87.98	84.79
BULGARIAN(19)	94.02	89.87	93.88	89.63
JAPANESE(8)	93.26	91.67	93.16	91.51
AVG	87.76	82.08	87.87	82.20

Table 2: Results for languages from CoNLL 2006/2007 shared tasks. When a language is in both years, the 2006 set is used. Languages are sorted by the number of unique arc labels.

guages times two primary beams – the best result is obtained by choosing a secondary beam size that is close to one half the size of the primary beam. Contrasting Table 1 and Table 3, the accuracy improvements are consistent across the development set and the test set for all three data sets.

A reasonable question is whether such improvements could be obtained by simply enlarging the beam in the baseline parser. The bottom row of Table 3 shows the parsing results for the three data sets when the beam is enlarged to 16. On Penn-S-2.0.5, the baseline with beam 16 is at roughly the same speed as the highlighted best system with primary beam 6 and secondary beam 3. On CTB-5, the beam 16 baseline is 30% slower. Table 3 indicates that simply enlarging the beam – relative to parsing speed – does not recover the wins of structural diversity on Penn-S-2.0.5 and CTB-5, though it does reduce the gap on Penn-S-2.0.5. On Penn-YM, the beam 16 baseline is slightly better than the new system, but 90% slower.

<i>primary beam</i>	<i>secondary beam</i>	PENN-YM		PENN-S-2.0.5		CTB-5	
		<i>UAS</i>	<i>LAS</i>	<i>UAS</i>	<i>LAS</i>	<i>UAS</i>	<i>LAS</i>
4	1	93.67	92.64	93.65	91.04	87.53	85.85
	2	93.79	92.68	93.77	91.30	87.62	85.96
	3	93.80	92.66	93.69	91.23	87.48	85.91
	4	93.75	92.63	93.62	91.11	87.68	86.08
6	1	93.65	92.46	93.76	91.15	87.72	86.05
	2	93.80	92.69	93.80	91.35	87.61	85.96
	3	93.75	92.64	93.99	91.55	87.80	86.18
	4	93.82	92.74	93.84	91.40	87.91	86.28
	5	93.82	92.71	93.71	91.26	87.75	86.12
	6	93.74	92.61	93.70	91.21	87.66	86.05
16	16	93.87	92.75	93.77	91.35	87.59	85.86

Table 3: Varying the degree of diversity by adjusting the secondary beam for labeled variants, with different primary beams. When the size of the secondary beam is equal to the primary beam, the parser degenerates to not enforcing structural diversity. In the opposite, when the secondary beam is smaller, there is more structural diversity and less label diversity. Results are on development sets.

To better understand the behaviour of structural diversity pruning relative to increasing the beam, we looked at the unlabeled attachment F-score per dependency label in the Penn-S-2.0.5 development set². Table 4 shows the 10 labels with the largest increase in attachment scores for structural diversity pruning relative to standard pruning. Importantly, the biggest wins are primarily for labels in which unlabeled attachment is lower than average (93.99, 8 out of 10). Thus, diversity pruning gets most of its wins on difficult attachment decisions. Indeed, many of the relations represent clausal dependencies that are frequently structurally ambiguous. There are also cases of relatively short dependencies that can be difficult to attach. For instance, *quantmod* dependencies are typically adverbs occurring after verbs that modify quantities to their right. But these can be confused as adverbial modifiers of the verb to the left. These results support our hypothesis that label ambiguity is causing hard attachment decisions to be pruned and that structural diversity can ameliorate this.

4 Discussion

Keeping multiple beams in approximate search has been explored in the past. In machine translation, multiple beams are used to prune translation hypotheses at different levels of granularity (Zens and Ney, 2008). However, the focus is improving the speed of translation decoder rather than improving translation quality through enforcement of hypothesis diversity. In parsing, Bohnet and Nivre (2012) and Bohnet et al. (2013) propose a model for joint morphological analysis, part-of-speech tagging and dependency parsing using a

<i>Label</i>	<i>w/o diversity large beam</i>	<i>w/ diversity small beam</i>	<i>diff</i>
quantmod	86.65	88.06	1.41
partmod	83.63	85.02	1.39
xcomp	87.76	88.74	0.98
tmod	89.75	90.72	0.97
appos	88.89	89.84	0.95
nsubjpass	92.53	93.31	0.78
complm	94.50	95.15	0.64
advcl	81.10	81.74	0.63
ccomp	82.64	83.17	0.54
number	96.86	97.39	0.53

Table 4: Unlabeled attachment F-score per dependency relation. The top 10 score increases for structural diversity pruning (beam 6 and label beam of 3) over basic pruning (beam 16) are shown. Only labels with more than 100 instances in the development data are considered.

left-to-right beam. With a single beam, token level ambiguities (morphology and tags) dominate and dependency level ambiguity is suppressed. This is addressed by essentially keeping two beams. The first forces every tree to be different at the dependency level and the second stores the remaining highest scoring options, which can include outputs that differ only at the token level.

The present work looks at beam diversity in graph-based dependency parsing, in particular label versus structural diversity. It was shown that by keeping a diverse beam significant improvements could be achieved on standard benchmarks, in particular with respect to difficult attachment decisions. It is worth pointing out that other dependency parsing frameworks (e.g., transition-based parsing (Zhang and Clark, 2008; Zhang and Nivre, 2011)) could also benefit from modeling structural diversity in search.

²Using eval.pl from Buchholz and Marsi (2006).

References

- S. Bergsma and C. Cherry. 2010. Fast and accurate arc filtering for dependency parsing. In *Proc. of COLING*.
- B. Bohnet and J. Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proc. of EMNLP/CoNLL*.
- B. Bohnet, J. Nivre, I. Boguslavsky, F. Ginter, Richárd F., and J. Hajic. 2013. Joint morphological and syntactic analysis for richly inflected languages. *TACL*, 1.
- S. Buchholz and E. Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proc. of CoNLL*.
- X. Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proc. of the CoNLL Shared Task Session of EMNLP-CoNLL*.
- D. Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2).
- M. De Marneffe, B. MacCartney, and C.D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proc. of LREC*.
- J. Eisner. 1996. Three new probabilistic models for dependency parsing: an exploration. In *Proc. of COLING*.
- H. He, H. Daumé III, and J. Eisner. 2013. Dynamic feature selection for dependency parsing. In *Proc. of EMNLP*.
- L. Kong and N. A. Smith. 2014. An empirical comparison of parsing methods for stanford dependencies. In *ArXiv:1404.4314*.
- T. Koo and M. Collins. 2010. Efficient third-order dependency parsers. In *Proc. of ACL*.
- S. Kübler, R. McDonald, and J. Nivre. 2009. *Dependency parsing*. Morgan & Claypool Publishers.
- X. Ma and H. Zhao. 2012. Fourth-order dependency parsing. In *Proc. of COLING*.
- A. F. T. Martins, M. B. Almeida, and N. A. Smith. 2013. Turning on the turbo: Fast third-order non-projective turbo parsers. In *Proc. of ACL*.
- R. McDonald and F. Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proc. of EACL*.
- R. McDonald, K. Crammer, and F. Pereira. 2005. Online large-margin training of dependency parsers. In *Proc. of ACL*.
- J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proc. of EMNLP-CoNLL*.
- S. Petrov, L. Barrett, R. Thibaux, and D. Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proc. of ACL*.
- A. Rush and S. Petrov. 2012. Efficient multi-pass dependency pruning with vine parsing. In *Proc. of NAACL*.
- R. Zens and H. Ney. 2008. Improvements in dynamic programming beam search for phrase-based statistical machine translation. In *Proc. IWSLT*.
- Y. Zhang and S. Clark. 2008. A Tale of Two Parsers: Investigating and Combining Graph-based and Transition-based Dependency Parsing. In *Proc. of EMNLP*.
- H. Zhang and R. McDonald. 2012. Generalized higher-order dependency parsing with cube pruning. In *Proc. of EMNLP*.
- Y. Zhang and J. Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proc. of ACL-HLT*, volume 2.
- H. Zhang, L. Huang, K. Zhao, and R. McDonald. 2013. Online learning for inexact hypergraph search. In *Proc. of EMNLP*.

The Penn Parsed Corpus of Modern British English: First Parsing Results and Analysis

Seth Kulick

Linguistic Data Consortium
University of Pennsylvania
skulick@ldc.upenn.edu

Anthony Kroch and Beatrice Santorini

Dept. of Linguistics
University of Pennsylvania
{kroch,beatrice}@ling.upenn.edu

Abstract

This paper presents the first results on parsing the Penn Parsed Corpus of Modern British English (PPCMBE), a million-word historical treebank with an annotation style similar to that of the Penn Treebank (PTB). We describe key features of the PPCMBE annotation style that differ from the PTB, and present some experiments with tree transformations to better compare the results to the PTB. First steps in parser analysis focus on problematic structures created by the parser.

1 Introduction

We present the first parsing results for the Penn Parsed Corpus of Modern British English (PPCMBE) (Kroch et al., 2010), showing that it can be parsed at a few points lower in F-score than the Penn Treebank (PTB) (Marcus et al., 1999). We discuss some of the differences in annotation style and source material that make a direct comparison problematic. Some first steps at analysis of the parsing results indicate aspects of the annotation style that are difficult for the parser, and also show that the parser is creating structures that are not present in the training material.

The PPCMBE is a million-word treebank created for researching changes in English syntax. It covers the years 1700-1914 and is the most modern in the series of treebanks created for historical research.¹ Due to the historical nature of the PPCMBE, it shares some of the characteristics of treebanks based on modern unedited text (Bies et al., 2012), such as spelling variation.

¹The other treebanks in the series cover Early Modern English (Kroch et al., 2004) (1.8 million words), Middle English (Kroch and Taylor, 2000) (1.2 million words), and Early English Correspondence (Taylor et al., 2006) (2.2 million words).

The size of the PPCMBE is roughly the same as the WSJ section of the PTB, and its annotation style is similar to that of the PTB, but with differences, particularly with regard to coordination and NP structure. However, except for Lin et al. (2012), we have found no discussion of this corpus in the literature.² There is also much additional material annotated in this style, increasing the importance of analyzing parser performance on this annotation style.³

2 Corpus description

The PPCMBE⁴ consists of 101 files, but we leave aside 7 files that consist of legal material with very different properties than the rest of the corpus. The remaining 94 files contain 1,018,736 tokens (words).

2.1 Part-of-speech tags

The PPCMBE uses a part-of-speech (POS) tag set containing 248 POS tags, in contrast to the 45 tags used by the PTB. The more complex tag set is mainly due to the desire to tag orthographic variants consistently throughout the series of historical corpora. For example “gentlemen” and its orthographic variant “gen’l’men” are tagged with the complex tag ADJ+NS (adjective and plural noun) on the grounds that in earlier time periods, the lexical item is spelled and tagged as two orthographic words (“gentle”/ADJ and “men”/NS).

While only 81 of the 248 tags are “simple” (i.e., not associated with lexical merging or splitting),

²Lin et al. (2012) report some results on POS tagging using their own mapping to different tags, but no parsing results.

³Aside from the corpora listed in fn. 1, there are also historical corpora of Old English (Taylor et al., 2003), Icelandic (Wallenberg et al., 2011), French (Martineau and others, 2009), and Portuguese (Galves and Faria, 2010), totaling 4.5 million words.

⁴We are working with a pre-release copy of the next revision of the official version. Some annotation errors in the currently available version have been corrected, but the differences are relatively minor.

Type	# Tags	# Tokens	% coverage
Simple	81	1,005,243	98.7%
Complex	167	13,493	1.3%
Total	248	1,018,736	100.0%

Table 1: Distribution of POS tags. Complex tags indicate lexical merging or splitting.

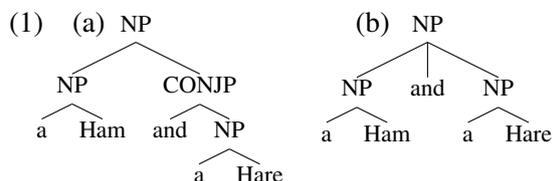


Figure 1: Coordination in the PPCMBE (1a) and the PTB (1b).

they cover the vast majority of the words in the corpus, as summarized in Table 1. Of these 81 tags, some are more specialized than in the PTB, accounting for the increased number of tags compared to the PTB. For instance, for historical consistency, words like “one” and “else” each have their own tag.

2.2 Syntactic annotation

As mentioned above, the syntactic annotation guidelines do not differ radically from those of the PTB. There are some important differences, however, which we highlight in the following three subsections.

2.2.1 Coordination

A coordinating conjunction and conjunct form a CONJP, as shown in (1a) in Figure 1. (1b) shows the corresponding annotation in the PTB.

In a conjoined NP, if part of a first conjunct potentially scopes over two or more conjuncts (shared pre-modifiers), the first conjunct has no phrasal node in the PPCMBE, and the label of the

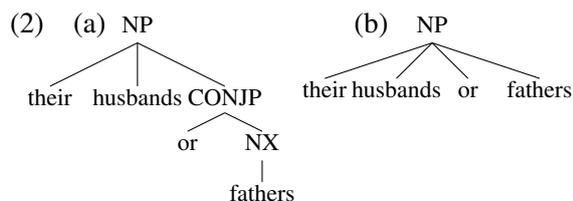


Figure 2: (2a) is an example of coordination with a shared pre-modifier in the PPCMBE, and (2b) shows the corresponding annotation in the PTB.

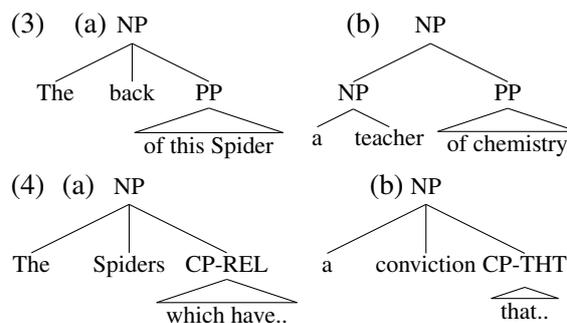


Figure 3: (3a) shows that a PP is sister to the noun in the PPCMBE, in contrast to the adjunction structure in the PTB (3b). (4a) and (4b) show that clausal complements and modifiers of a noun are distinguished by function tags, rather than structurally as in the PTB, which would adjoin the CP in (a), but not in (b).

subsequent conjuncts becomes NX instead of NP, as shown in (2a) in Figure 2. The corresponding PTB annotation is flat, as in (2b).⁵

2.2.2 Noun Phrase structure

Neither the PPCMBE nor the PTB distinguish between PP complements and modifiers of nouns. However, the PPCMBE annotates both types of dependents as sisters of the noun, while the PTB adjoins both types. For instance in (3a) in Figure 3, the modifier PP is a sister to the noun in the PPCMBE, while in (3b), the complement PP is adjoined in the PTB.

Clausal complements and modifiers are also both treated as sisters to the noun in the PPCMBE. In this case, though, the complement/modifier distinction is encoded by a function tag. For example, in (4a) and (4b), the status of the CPs as modifier and complement is indicated by their function tags: REL for relative clause and THT “that” complement. In the PTB, the distinction would be encoded structurally; the relative clause would be adjoined, whereas the “that” complement would not.

2.2.3 Clausal structure

The major difference in the clausal structure as compared to the PTB is the absence of a VP level⁶, yielding flatter trees than in the PTB. An example clause is shown in (5) in Figure 4.

⁵Similar coordination structures exist for categories other than NP, although NP is by far the most common.

⁶This is due to the changing headedness of VP in the overall series of English historical corpora.

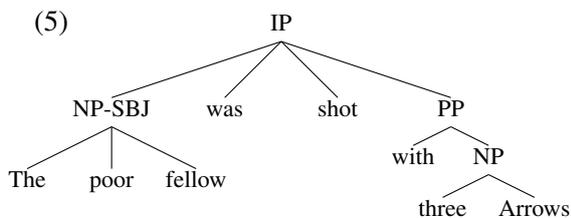


Figure 4: An example of clausal structure, without VP.

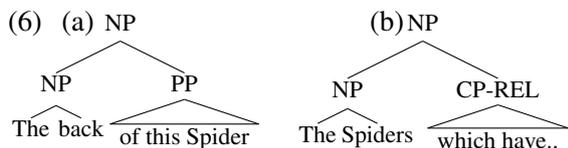


Figure 5: (6a) shows how (3a) is transformed in the “reduced +NPs” version to include a level of NP recursion, and (6b) shows the same for (4a).

3 Corpus transformations

We refer to the pre-release version of the corpus described in Section 2 as the “Release” version, and experiment with three other corpus versions.

3.1 Reduced

As mentioned earlier, the PPCMBE’s relatively large POS tag set aims to maximize annotation consistency across the entire time period covered by the historical corpora, beginning with Middle English. Since we are concerned here with parsing just the PPCMBE, we simplified the tag set.

The complex tags are simplified in a fully deterministic way, based on the trees and the tags. For example, the POS tag for “gentleman”, originally ADJ+N is changed to N. The P tag is split, so that it is either left as P, if a preposition, or changed to CONJS, if a subordinating conjunction. The reduced tag set contains 76 tags. We call the version of the corpus with the reduced tag set the “Reduced” version.

3.2 Reduced+NPs

As discussed in Section 2.2.2, noun modifiers are sisters to the noun, instead of being adjoined, as in the PTB. As a result, there are fewer NP brackets in the PPCMBE than there would be if the PTB-style were followed. To evaluate the effect of the difference in annotation guidelines on the parsing score, we added PTB-style NP brackets to the reduced corpus described in Section 3.1. For example, (3a) in Figure 3 is transformed into (6a)

Section	# Files	Token count	%
Train	81	890,150	87.4%
Val	4	38,670	3.8%
Dev	4	39,527	3.9%
Test	5	50,389	4.9%
Total	94	1,018,736	100.0%

Table 2: Token count and data split for PPCMBE

in Figure 5, and likewise (4a) is transformed into (6b). However, (4b) remains as it is, because the following CP in that case is a complement, as indicated by the THT function tag. This is a significant transformation of the corpus, adding 43,884 NPs to the already-existing 291,422.

3.3 Reduced+NPs+VPs

We carry out a similar transformation to add VP nodes to the IPs in the Reduced+NPs version, making them more like the clausal structures in the PTB. This added 169,877 VP nodes to the corpus (there are 131,671 IP nodes, some of which contain more than one auxiliary verb).

It is worth emphasizing that the brackets added in Sections 3.2 and 3.3 add no information, since they are added automatically. They are added only to roughly compensate for the difference in annotation styles between the PPCMBE and the PTB.

4 Data split

We split the data into four sections, as shown in Table 2. The validation section consists of the four files beginning with “a” or “v” (spanning the years 1711-1860), the development section consists of the four files beginning with “l” (1753-1866), the test section consists of the five files beginning with “f” (1749-1900), and the training section consists of the remaining 81 files (1712-1913). The data split sizes used here for the PPCMBE closely approximate that used for the PTB, as described in Petrov et al. (2006).⁷ For this first work, we used a split that was roughly the same as far as time-spans across the four sections. In future work, we will do a more proper cross-validation evaluation.

Table 3 shows the average sentence length and percentage of sentences of length ≤ 40 in the PPCMBE and PTB. The PPCMBE sentences are a bit longer on average, and fewer are of length ≤ 40 . However, the match is close enough that

⁷Sections 2-21 for Training Section 1 for Val, 22 for Dev and 23 for Test.

	Corpus	Gold Tags						Parser Tags						Tags
		all			<=40			all			<=40			
		Prec	Rec	F	Prec	Rec	F	Prec	Rec	F	Prec	Rec	F	
1	RI/Dev	83.7	83.7	83.7	86.3	86.4	86.3	83.8	83.1	83.4	86.2	85.8	86.0	96.9
2	Rd/Dev	84.9	84.5	84.7	86.6	86.7	86.7	84.5	83.7	84.1	86.5	86.2	86.3	96.9
3	Rd/Tst	85.8	85.2	85.5	87.9	87.3	87.6	84.8	83.9	84.3	86.7	85.8	86.2	97.1
4	RdNPs/Dev	87.1	86.3	86.7	88.9	88.5	88.7	86.3	85.1	85.7	88.4	87.6	88.0	96.9
5	RdNPsVPs/Dev	87.2	87.0	87.1	89.5	89.4	89.5	86.3	85.7	86.0	88.6	88.2	88.4	97.0
6	PTB/23	90.3	89.8	90.1	90.9	90.4	90.6	90.0	89.5	89.8	90.6	90.1	90.3	96.9

Table 4: Parsing results with Berkeley Parser. The corpus versions used are Release (RI), Reduced (Rd), Reduced+NPs (RdNPs), and Reduced+NPs+VPs (RdNPsVPs). Results are shown for the parser forced to use the gold POS tags from the corpus, and with the parser supplying its own tags. For the latter case, the tagging accuracy is shown in the last column.

Corpus	Section	Avg. len	% <= 40
PPCMBE	Dev	24.1	85.5
	Test	21.2	89.9
PTB	Dev	23.6	92.9
	Test	23.5	91.3

Table 3: Average sentence length and percentage of sentences of length ≤ 40 in the PPCMBE and PTB.

we will report the parsing results for sentences of length ≤ 40 and all sentences, as with the PTB.

5 Parsing Experiments

The PPCMBE is a phrase-structure corpus, and so we parse with the Berkeley parser (Petrov et al., 2008) and score using the standard evalb program (Sekine and Collins, 2008). We used the Train and Val sections for training, with the parser using the Val section for fine-tuning parameters (Petrov et al., 2006). Since the Berkeley parser is capable of doing its own POS tagging, we ran it using the gold tags or supplying its own tags. Table 4 shows the results for both modes.⁸

Consider first the results for the Dev section with the parser using the gold tags. The score for all sentences increases from 83.7 for the Release corpus (row 1) to 84.7 for the Reduced corpus (row 2), reflecting the POS tag simplifications in the Reduced corpus. The score goes up by a further 2.0 to 86.7 (row 2 to 4) for the Reduced+NPs corpus and up again by 0.4 to 87.1 (row 5) for the Reduced+NPs+VPs corpus, showing the ef-

⁸We modified the evalb parameter file to exclude punctuation in PPCMBE, just as for PTB. The results are based on a single run for each corpus/section. We expect some variance to occur, and in future work will average results over several runs of the training/Dev cycle, following Petrov et al. (2006).

fects of the extra NP and VP brackets. We evaluated the Test section on the Reduced corpus (row 3), with a result 0.8 higher than the Dev (85.5 in row 3 compared to 84.7 in row 2). The score for sentences of length ≤ 40 (a larger percentage of the PPCMBE than the PTB) is 2.4 higher than the score for all sentences, with both the gold and parser tags (row 5).

The results with the parser choosing its own POS tags naturally go down, with the Test section suffering more. In general, the PPCMBE is affected by the lack of gold tags more than the PTB.

In sum, the parser results show that the PPCMBE can be parsed at a level approaching that of the PTB. We are not proposing that the current version be replaced by the Reduced+NPs+VPs version, on the grounds that the latter gets the highest score. Our goal was to determine whether the parsing results fell in the same general range as for the PTB by roughly compensating for the difference in annotation style. The results in Table 4 show that this is the case.

As a final note, the PPCMBE consists of unedited data spanning more than 200 years, while the PTB is edited newswire, and so to some extent there would almost certainly be some difference in score.

6 Parser Analysis

We are currently developing techniques to better understand the types of errors is making, which have already led to interesting results. The parser is creating some odd structures that violate basic well-formedness conditions of clauses. Tree (7a) in Figure 6 is a tree from from the ‘‘Reduced’’ corpus, in which the verb ‘‘formed’’ projects to IP,

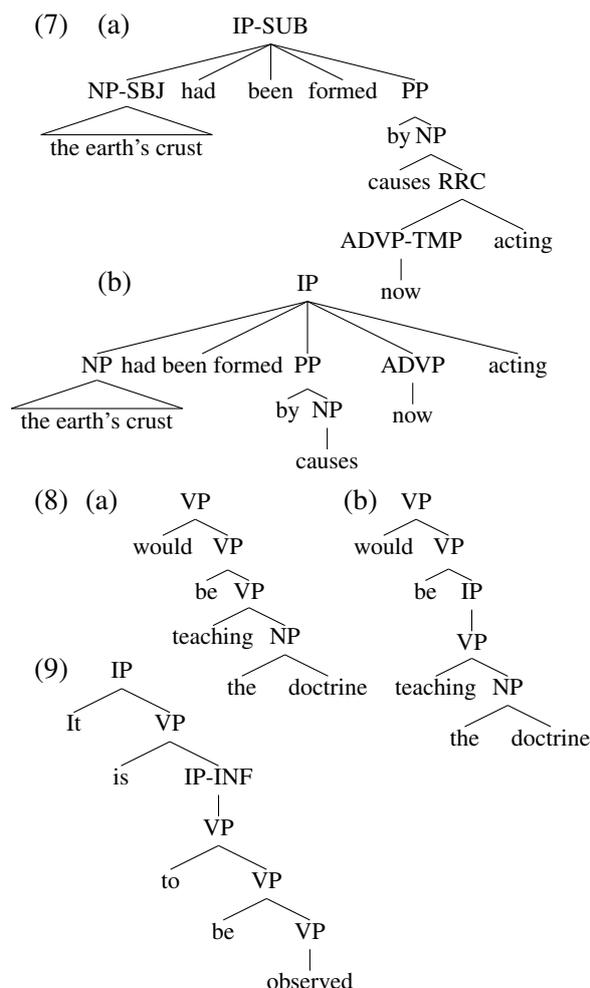


Figure 6: Examples of issues with parser output

with two auxiliary verbs (“had” and “been”). In the corresponding parser output (7b), the parser misses the reduced relative RRC, turning “acting” into the rightmost verb in the IP. The parser is creating an IP with two main verbs - an ungrammatical structure that is not attested in the gold.

It might be thought that the parser is having trouble with the flat-IP annotation style, but the parser posits incorrect structures that are not attested in the gold even in the Reduced+NPs+VPs version of the corpus. Tree (8a) shows a fragment of a gold tree from the corpus, with the VPs appropriately inserted. The parser output (8b) has an extra IP above “teaching”. The POS tags for “be” (BE) and “teaching” (VAG) do not appear in this configuration at all in the training material. In general, the parser seems to be getting confused as to when such an IP should appear. We hypothesized that this is due to confusion with infinitival clauses, which can have an unary-branching IP

over a VP, as in the gold tree (9). We retrained the parser, directing it to retain the INF function tag that appears in infinitival clauses as in (9). Overall, the evalb score went down slightly, but it did fix cases such as (8b). We do not yet know why the overall score went down, but what’s surprising is one would have thought that IP-INF is recoverable from the absence of a tensed verb.

Preliminary analysis shows that the CONJP structures are also difficult for the parser. Since these are structures that are different than the PTB⁹, we were particularly interested in them. Cases where the CONJP is missing an overt coordinating cord (such as “and”), are particularly difficult, not surprisingly. These can appear as intermediate conjuncts in a string of conjuncts, with the structure (CONJP word). The shared pre-modifier structure described in (2a) is also difficult for the parser.

7 Conclusion

We have presented the first results on parsing the PPCMBE and discussed some significant annotation style differences from the PTB. Adjusting for two major differences that are a matter of annotation convention, we showed that the PPCMBE can be parsed at approximately the same level of accuracy as the PTB. The first steps in an investigation of the parser differences show that the parser is generating structures that violate basic well-formedness conditions of the annotation.

For future work, we will carry out a more serious analysis of the parser output, trying to more properly account for the differences in bracketing structure between the PPCMBE and PTB. There is also a great deal of data annotated in the style of the PPCMBE, as indicated in footnotes 1 and 3, and we are interested in how the parser performs on these, especially comparing the results on the modern English corpora to the older historical ones, which will have greater issues of orthographic and tokenization complications.

Acknowledgments

This work was supported by National Science Foundation Grant # BCS-114749. We would like to thank Ann Bies, Justin Mott, and Mark Liberman for helpful discussions.

⁹The CONJP nonterminal in the PTB serves a different purpose than in the PPCMBE and is much more limited.

References

- Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. English Web Treebank. LDC2012T13. Linguistic Data Consortium.
- Charlotte Galves and Pabol Faria. 2010. Tycho Brahe Parsed Corpus of Historical Portuguese. <http://www.tycho.iel.unicamp.br/~tycho/corpus/en/index.html>.
- Anthony Kroch and Ann Taylor. 2000. Penn-Helsinki Parsed Corpus of Middle English, second edition. <http://www.ling.upenn.edu/hist-corpora/PPCME2-RELEASE-3/index.html>.
- Anthony Kroch, Beatrice Santorini, and Ariel Dierani. 2004. Penn-Helsinki Parsed Corpus of Early Modern English. <http://www.ling.upenn.edu/hist-corpora/PPCEME-RELEASE-2/index.html>.
- Anthony Kroch, Beatrice Santorini, and Ariel Dierani. 2010. Penn Parsed Corpus of Modern British English. <http://www.ling.upenn.edu/hist-corpora/PPCMBE-RELEASE-1/index.html>.
- Yuri Lin, Jean-Baptiste Michel, Erez Aiden Lieberman, Jon Orwant, Will Brockman, and Slav Petrov. 2012. Syntactic annotations for the google books ngram corpus. In *Proceedings of the ACL 2012 System Demonstrations*, pages 169–174, Jeju Island, Korea, July. Association for Computational Linguistics.
- Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. Treebank-3. LDC99T42, Linguistic Data Consortium, Philadelphia.
- France Martineau et al. 2009. Modéliser le changement: les voies du français, a Parsed Corpus of Historical French.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of COLING-ACL*, pages 433–440, Sydney, Australia, July. Association for Computational Linguistics.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2008. The Berkeley Parser. <https://code.google.com/p/berkeleyparser/>.
- Satoshi Sekine and Michael Collins. 2008. Evalb. <http://nlp.cs.nyu.edu/evalb/>.
- Ann Taylor, Anthony Warner, Susan Pintzuk, and Frank Beths. 2003. The York-Toronto-Helsinki Parsed Corpus of Old English Prose. Distributed through the Oxford Text Archive. <http://www-users.york.ac.uk/~lang22/YCOE/YcoeHome.htm>.
- Ann Taylor, Arja Nurmi, Anthony Warner, Susan Pintzuk, and Terttu Nevalainen. 2006. Parsed Corpus of Early English Correspondence. Compiled by the CEEC Project Team. York: University of York and Helsinki: University of Helsinki. Distributed through the Oxford Text Archive. <http://www-users.york.ac.uk/~lang22/PCEEC-manual/index.htm>.
- Joel Wallenberg, Anton Karl Ingason, Einar Freyr Sigursson, and Eirkur Rgnvaldsson. 2011. Icelandic Parsed Historical Corpus (IcePaHC) version 0.4. http://www.linguist.is/icelandic_treebank.

Parser Evaluation Using Derivation Trees: A Complement to evalb

Seth Kulick and Ann Bies and Justin Mott

Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA 19104
{skulick,bies,jmott}@ldc.upenn.edu

Anthony Kroch and Mark Liberman and Beatrice Santorini

Department of Linguistics, University of Pennsylvania, Philadelphia, PA 19104
{kroch,myl,beatrice}@ling.upenn.edu

Abstract

This paper introduces a new technique for phrase-structure parser analysis, categorizing possible treebank structures by integrating regular expressions into derivation trees. We analyze the performance of the Berkeley parser on OntoNotes WSJ and the English Web Treebank. This provides some insight into the evalb scores, and the problem of domain adaptation with the web data. We also analyze a “test-on-train” dataset, showing a wide variance in how the parser is generalizing from different structures in the training material.

1 Introduction

Phrase-structure parsing is usually evaluated using evalb (Sekine and Collins, 2008), which provides a score based on matching brackets. While this metric serves a valuable purpose in pushing parser research forward, it has limited utility for understanding what sorts of errors a parser is making. This is the case even if the score is broken down by brackets (NP, VP, etc.), because the brackets can represent different types of structures. We would also like to have answers to such questions as “How does the parser do on non-recursive NPs, separate from NPs resulting from modification? On PP attachment?” etc.

Answering such questions is the goal of this work, which combines two strands of research. First, inspired by the tradition of Tree Adjoining Grammar-based research (Joshi and Schabes, 1997; Bangalore and Joshi, 2010), we use a decomposition of the full trees into “elementary trees” (henceforth “etrees”), with a derivation tree that records how the etrees relate to each other, as in Kulick et al. (2011). In particular, we use the “spinal” structure approach of (Shen et al., 2008; Shen and Joshi, 2008), where etrees are constrained to be unary-branching.

Second, we use a set of regular expressions (henceforth “regexes”) that categorize the possible structures in the treebank. These are best thought of as an extension of head-finding rules, which not only find a head but simultaneously identify each parent/children relation as one of a limited number of types of structures (right-modification, etc.).

The crucial step is that we integrate these regexes into the spinal etrees. The derivation trees provide elements of a dependency analysis, which allow us to calculate scores for head identification and attachment for different projections (e.g., PP). The regexes allow us to also provide scores based on spans of different construction types. Together these two aspects break down the evalb brackets into more meaningful categories, and the simultaneous head and span scoring allows us to separate these aspects in the analysis.

After describing in more detail the basic framework, we show some aspects of the resulting analysis of the performance of the Berkeley parser (Petrov et al., 2008) on three datasets: (a) OntoNotes WSJ sections 2-21 (Weischedel et al., 2011)¹, (b) OntoNotes WSJ section 22, and (c) the “Answers” section of the English Web Treebank (Bies et al., 2012). We trained the parser on sections 2-21, and so (a) is “test-on-train”. These three results together show how the parser is generalizing from the training data, and what aspects of the “domain adaptation” problem to the web material are particularly important.²

2 Framework for analyzing parsing performance

We first describe the use of the regexes in tree decomposition, and then give some examples of in-

¹We refer only to the WSJ treebank portion of OntoNotes, which is roughly a subset of the Penn Treebank (Marcus et al., 1999) with annotation revisions including the addition of NML nodes.

²We parse (c) while training on (a) to follow the procedure in Petrov and McDonald (2012)

incorporating these regexes into the derivation trees.

2.1 Use of regular expressions

Decomposing the original phrase-structure tree into the smaller components requires some method of determining the “head” of a nonterminal, from among its children nodes, similar to parsing work such as Collins (1999). As described above, we are also interested in the type of linguistic construction represented by that one-level structure, each of which instantiates one of a few types - recursive coordination, simple head-and-sister, etc. We address both tasks together with the regexes. In contrast to the sort of head rules in (Collins, 1999), these refer as little as possible to specific POS tags. Instead of explicitly listing the POS tags of possible heads, the heads are in most cases determined by their location in the structure.

Sample regexes are shown in Figure 1. There are 49 regexes used.³ Regexes ADJP-t and ADVP-t in (a) identify their terminal head to be the rightmost terminal, possibly preceded by some number of terminals or nonterminals, relying on a mapping that maps all terminals (except CC, which is mapped to CONJ) to TAG and all nonterminals (except CONJP and NML) to NT. Structures with a CONJ/CONJP/NML child do not match this rule and are handled by different regexes, which are all mutually exclusive. In some cases, we need to search for particular nonterminal heads, such as with the (b) regexes S-vp and SQ-vp, which identify the rightmost VP among the children of a S or SQ as the head. (c) NP-modr is a regex for a recursive NP with a right modifier. In this case, the NP on the left is identified as the head. (d) VP-crd is also a regex for a recursive structure, in this case for VP coordination, picking out the leftmost conjunct as the head of the structure. The regex names roughly describe their purpose - “mod” for right-modification, “crd” for coordination, etc. The suffix “-t” is for the simple non-recursive case in which the head is a terminal.

2.2 Regexes in the derivation trees

The names of these regexes are incorporated into the etrees themselves, as labels of the nonterminals. This allows an etree to contain information

³Some among the 49 are duplicates, used for different nonterminals, as with (a) and (b) in Figure 1. We derived the regexes via an iterative process of inspection of tree decomposition on dataset (a), together with taking advantage of the treebanking experience from some of the co-authors.

```
(a)ADJP-t,ADVP-t:
^(TAG|NT|NML)*(head:TAG) (NT)*$
(b)S-vp, SQ-vp: ^([\^ ]+)*(head:VP)$
(c)NP-modr:
^(head:NP) (SBAR|S|VP|ADJP|PP|ADVP|NP)+$
(d)VP-crd: ^ (head:VP) (VP)* CONJ VP$
```

Figure 1: Some sample regexes

such as “this node represents right modification”.

For example, Figure 2 shows the derivation tree resulting from the decomposition of the tree in Figure 4. Each structure within a circle is one etree, and the derivation as a whole indicates how these etrees are combined. Here we indicate with arrows that point to the relevant regex. For example, the PP-t etree #a6 points to the NP-modr regex, which consists of the NP-t together with the PP-t. The nonterminals of the spinal etrees are the names of the regexes, with the simpler non-terminal labels trivially derivable from the regex names.⁴

The tree in Figure 5 is the parser output corresponding to the gold tree in Figure 4, and in this case gets the PP-t attachment wrong, while everything else is the same as the gold.⁵ This is reflected in the derivation tree in Figure 3, in which the NP-modr regex is absent, with the NP-t and PP-t etrees #b5 and #b6 both pointing to the VP-t regex in #b3. We show in Section 2.3 how this derivation tree representation is used to score this attachment error directly, rather than obscuring it as an NP bracket error as evalb would do.

2.3 Scoring

We decompose both the gold and parser output trees into derivation trees with spinal etrees, and score based on the regexes projected by each word. There is a match for a regex if the corresponding words in gold/parser files project to that regex, a precision error if the parser file does but the gold does not, and a recall error if the gold does but the parser file does not.

For example, comparing the trees in Figures 4 and 5 via their derivation trees in Figures 2 and Figures 3, the word “trip” has a match for the regex NP-t, but a recall error for NP-modr. The word

⁴We do not have space here to discuss the data structure in complete detail, but multiple regex names at a node, such a VP-aux and VP-t at tree a3 in Figure 2, indicate multiple VP nonterminals.

⁵We leave function tags aside for future work. The gold tree is shown without the SBJ function tag.

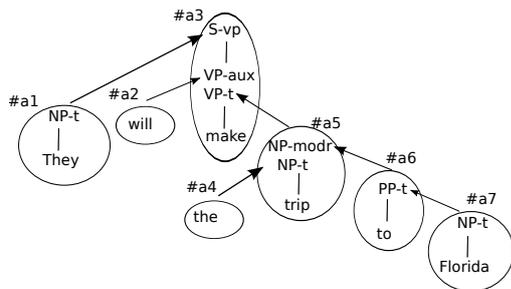


Figure 2: Derivation Tree for Figure 4

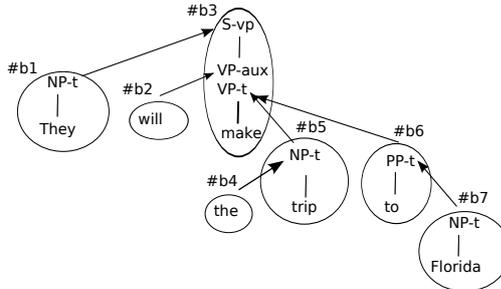


Figure 3: Derivation Tree for Figure 5)

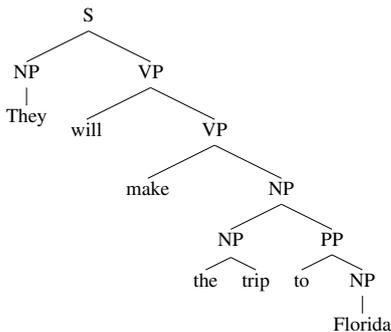


Figure 4: Gold tree

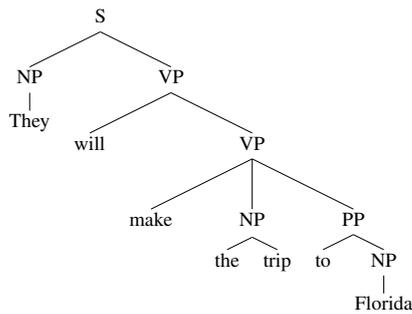


Figure 5: Parser output tree

Corpus	tokens	brackets	coverage	%	evalb
2-21 g	650877	578597	571243	98.7	
p		575744	569480	98.9	93.8
22 g	32092	24819	24532	98.8	
p		24801	24528	98.9	90.1
Ans g	53960	48492	47348	97.6	
p		48750	47423	97.3	80.8

Table 1: Corpus information for gold(g) and parsed(p) sections of each corpus

“make” has a match for the regexes VP-t, VP-aux, and S-vp, and so on. Summing such scores over the corresponding gold/parser trees gives us F-scores for each regex.

There are two modifications/extensions to these F-scores that we also use:

(1) For each regex match, we score whether it matches based on the span as well. For example, “make” is a match for VP-t in Figures 2 and 3, and is also a match for the span as well, since in both derivation trees it includes the words “make...Florida”. It is this matching for span as well as head that allows us to compare our results to evalb. We call the match just for the head the “F-h” score and the match that also includes the span information the “F-s” score. The F-s score roughly corresponds to the evalb score. However, the F-

s score is for separate syntactic constructions (including also head identification), although we can also sum it over all the structures, as done later in Figure 6. The simultaneous F-h and F-s scores let us identify constructions where the parser has the head projection correct, but gets the span wrong. (2) Since the derivation tree is really a dependency tree with more complex nodes (Rambow and Joshi, 1997; Kulick et al., 2012), we can also score each regex for attachment.⁶ For example, while “to” is a match for PP-t, its attachment is not, since in Figure 2 it is a child of the “trip” etree (#a5) and in Figure 3 it is a child of the “make” etree (#b3). Therefore our analysis results in an attachment score for every regex.

2.4 Comparison with previous work

This work is in the same basic line of research as the inter-annotator agreement analysis work in Kulick et al. (2013). However, that work did not utilize regexes, and focused on comparing sequences of identical strings. The current work scores on general categories of structures, without

⁶A regex intermediate in a etree, such as VP-t above, is considered to have a default null attachment. Also, the attachment score is not relevant for regexes that already express a recursive structure, such as NP-modr. In Figure 2, NP-t in etree #a5 is considered as having the attachment to #a3.

regex	Sections 2-21 (Ontonotes)					Section 22 (Ontonotes)					Answers (English Web Treebank)				
	%gold	F-h	F-s	att	spanR	%gold	F-h	F-s	att	spanR	%gold	F-h	F-s	att	spanR
NP-t	30.7	98.9	97.6	96.5	99.6	31.1	98.0	95.8	94.4	99.6	28.5	95.4	91.5	90.9	99.3
VP-t	13.5	98.8	94.5	98.4	95.8	13.4	98.1	91.7	97.3	93.7	16.0	96.7	81.7	96.1	85.4
PP-t	12.2	99.2	91.0	90.5	92.0	12.1	98.7	86.4	86.1	88.2	8.4	96.4	80.5	80.7	84.7
S-vp	12.2	97.9	92.8	96.8	96.3	11.9	96.5	89.1	95.4	95.0	14.2	94.1	72.9	88.0	84.1
NP-modr	8.6	88.4	80.3	-	91.5	8.5	82.9	71.8	-	87.9	4.4	69.0	54.2	-	80.5
VP-aux	5.5	97.9	94.0	-	96.1	5.0	96.5	91.0	-	94.6	6.2	94.4	81.7	-	86.7
SBAR-s	3.7	96.1	91.1	91.8	95.3	3.5	94.3	87.2	86.4	93.5	4.0	84.8	68.2	81.9	81.9
ADVP-t	2.7	95.2	93.3	93.9	98.6	3.0	89.6	84.5	88.0	95.9	4.5	84.0	78.2	80.3	96.8
NML-t	2.3	91.6	90.3	97.6	99.8	2.6	85.6	82.2	93.5	99.8	0.7	42.1	37.7	88.8	100.0
ADJP-t	1.9	94.6	88.4	95.5	94.6	1.8	86.8	77.0	93.6	90.7	2.5	84.7	67.0	88.1	84.2
QP-t	1.0	95.3	93.8	98.3	99.6	1.2	91.0	89.0	97.1	100.0	0.2	57.7	57.7	94.4	100.0
NP-crd	0.8	80.3	73.7	-	92.4	0.6	68.6	58.4	-	86.1	0.5	55.3	47.8	-	88.1
VP-crd	0.4	84.3	82.8	-	98.2	0.4	75.3	73.5	-	97.6	0.8	65.5	58.3	-	89.8
S-crd	0.3	83.7	83.2	-	99.6	0.4	70.9	68.6	-	96.7	0.8	68.5	63.0	-	93.4
SQ-v	0.1	88.3	82.0	93.3	97.8	0.1	66.7	66.7	88.9	100.0	0.9	81.9	72.4	93.4	95.8
FRAG-nt	0.1	49.9	48.6	95.4	97.9	0.1	28.6	28.6	100.0	100.0	0.8	22.7	21.3	96.3	96.3

Table 2: Scores for the most frequent categories of brackets in the three datasets of corpora, as determined by the regexes. % gold is the frequency of this regex type compared to all the brackets in the gold. F-h is the score based on matching heads, F-s also incorporates the span information, att is the attachment accuracy for words that match in F-h, and spanR is the span-right accuracy for words that match in F-h.

the reliance on sequences of individual strings.⁷

3 Analysis of parsing results

We worked with the three datasets as described in the introduction. We trained the parser on sections 2-21 of OntoNotes WSJ, and parsed the three datasets with the gold tags, since at present we wish to analyze the parser performance in isolation from Part-of-Speech tagging errors. Table 1 shows the sizes of the three corpora in terms of tokens and brackets, for both the gold and parsed versions, with the evalb scores for the parsed versions. The score is lower for Answers, as also found by Petrov and McDonald (2012).

To facilitate comparison of our analysis with evalb, we used corpora versions with the same bracket deletion (empty yields and most punctuation) as evalb. We ran the gold and parsed versions through our regex decomposition and derivation tree creation. Table 1 shows the number and percentage of brackets handled by our regexes. The high coverage (%) reinforces the point that there is a limited number of core structures in the treebank. In the results below in Table 2 and Figure 6 we combine the nonterminals that are not covered by one of the regexes with the simple non-recursive regex case for that nonterminal.⁸

⁷In future work we will compare our approach to that of Kummerfeld et al. (2012), who also move beyond evalb scores in an effort to provide more meaningful error analysis.

⁸We also combine a few other non-recursive regexes together with NP-t, such as the special one for possessives.

We present the results in two ways. Table 2 lists the most frequent categories in the three datasets, with their percentage of the overall number of brackets (%gold), their score based just on the head identification (F-h), their score based on head identification and (left and right) span (F-s), and the attachment (att) and span-right (spanR) scores for those that match based on the head.⁹

The two graphs in Figure 6 show the cumulative results based on F-h and F-s, respectively. These show the cumulative score in order of the frequency of categories. For example, for sections 2-21, the score for NP-t is shown first, with 30.7% of the brackets, and then together with the VP-t category, they cover 45.2% of the brackets, etc.¹⁰ The benefit of the approach described here is that now we can see the contribution to the evalb score of the particular types of constructions, and within those constructions, how well the parser is doing at getting the same head projection, but failing or

⁹The score for the left edge is almost always very high for every category, and we just list here the right edge score. The attachment score does not apply to the recursive categories, as mentioned above.

¹⁰The final F-s value is lower than the evalb score - e.g. 92.5 for sections 2-21 (the rightmost point in the graph for sections 2-21 in the F-s graph in Figure 6) compared to the 93.8 evalb score. Space prevents full explanation, but there are two reasons for this. One is that there are cases in which bracket spans match, but the head, as found by our regexes, is different in the gold and parser trees. The other cases is when brackets match, and may even have the same head, but their regex is different. In future work we will provide a full accounting of such cases, but they do not affect the main aspects of the analysis.

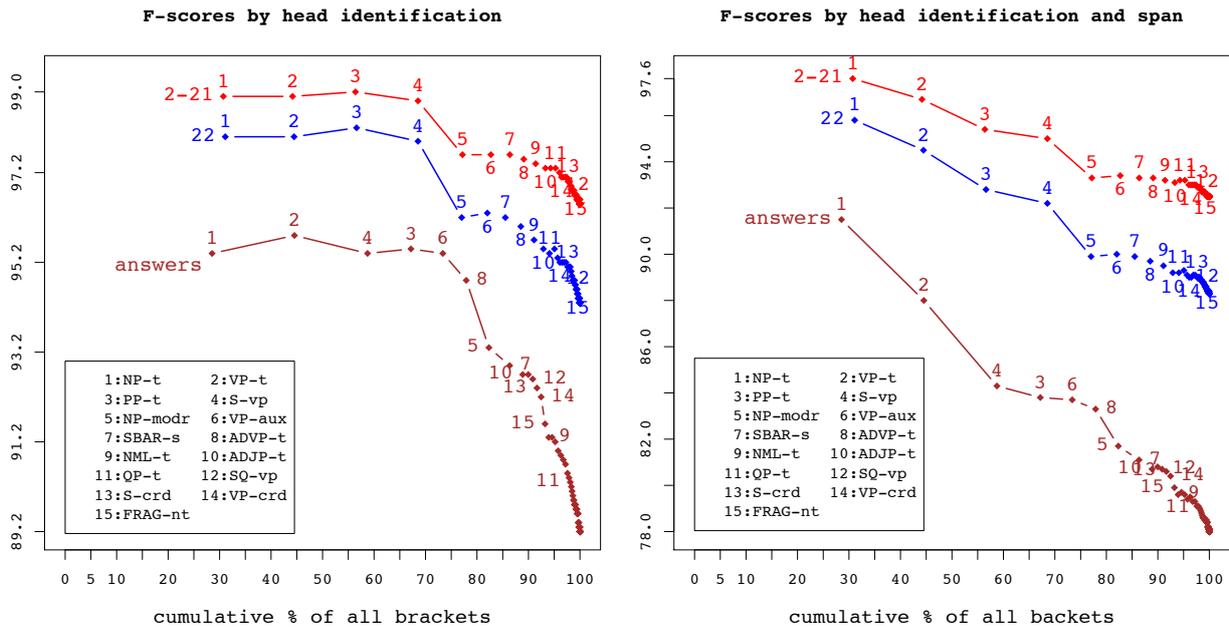


Figure 6: Cumulative scores based on F-h (left) and F-s (right). These graphs are both cumulative in exactly the same way, in that each point represents the total percentage of brackets accounted for so far. So for the 2-21 line, point 1, meaning the NP non-recursive regex, accounts for 30.7% of the brackets, point 2, meaning the VP non-recursive regex, accounts for another 13.5%, so 44.2% cumulatively, etc.

not on the spans.

3.1 Analysis and future work

As this is work-in-progress, the analysis is not yet complete. We highlight a few points here.

(1) The high performance on the OntoNotes WSJ material is in large part due to the score on the non-recursive regexes of NP-t, VP-t, S-vp, and the auxiliaries (points 1, 2, 4, 6 in the graphs). Critical to this is the fact that the parser does well on determining the right edge of verbal structures, which affects the F-s score for VP-t (non-recursive), VP-aux, and S-vp. The spanR score for VP-t is 95.8 for Sections 2-21 and 93.7 for Section 22.

(2) We wouldn't expect the test-on-training evalb score to be 100%, since it has to back off from the training data, but the results for the different categories vary widely, with e.g., the NP-modr F-h score much lower than other frequent regexes. This variance from the test-on-training dataset carries over almost exactly to Section 22.

(3) The different distribution of structures in Answers hurts performance. For example, the mediocre performance of the parser on SQ-vp barely affects the score with OntoNotes, but has a larger negative effect with Answers, due to its increased frequency in the latter.

(4) While the different distribution of construc-

tions is a problem for Answers, more critical is the poor performance of the parser on determining the right edge of verbal constructions. This is only 85.4 for VP-t in Answers, compared to the OntoNotes results mentioned in (1). Since this affects the F-s scores for VP-t, VP-aux, and S-vp, the negative effect is large. Preliminary investigation shows that this is due in part to incorrect PP and SBAR placement (the PP-t and SBAR-s attachment scores (80.7 and 81.9) are worse for Answers compared to Section 22 (86.1 and 86.4)), and coordinated S-clauses with no conjunction.

In sum, there is a wealth of information from this new type of analysis that we will use in our ongoing work to better understand what the parser is learning and how it works on different genres.

Acknowledgments

This material is based upon work supported by National Science Foundation Grant # BCS-114749 (first, fourth, and sixth authors) and by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-11-C-0145 (first, second, and third authors). The content does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred.

References

- Srinivas Bangalore and Aravind K. Joshi, editors. 2010. *Supertagging: Using Complex Lexical Descriptions in Natural Language Processing*. MIT Press.
- Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. English Web Treebank. LDC2012T13. Linguistic Data Consortium.
- Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, Department of Computer and Information Sciences, University of Pennsylvania.
- A.K. Joshi and Y. Schabes. 1997. Tree-adjointing grammars. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages, Volume 3: Beyond Words*, pages 69–124. Springer, New York.
- Seth Kulick, Ann Bies, and Justin Mott. 2011. Using derivation trees for treebank error detection. Association for Computational Linguistics.
- Seth Kulick, Ann Bies, and Justin Mott. 2012. Using supertags and encoded annotation principles for improved dependency to phrase structure conversion. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 305–314, Montréal, Canada, June. Association for Computational Linguistics.
- Seth Kulick, Ann Bies, Justin Mott, Mohamed Maamouri, Beatrice Santorini, and Anthony Kroch. 2013. Using derivation trees for informative treebank inter-annotator agreement evaluation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 550–555, Atlanta, Georgia, June. Association for Computational Linguistics.
- Jonathan K. Kummerfeld, David Hall, James R. Curran, and Dan Klein. 2012. Parser showdown at the wall street corral: An empirical investigation of error types in parser output. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1048–1059, Jeju Island, Korea, July. Association for Computational Linguistics.
- Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. Treebank-3. LDC99T42, Linguistic Data Consortium, Philadelphia.
- Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web. In *Proceedings of the First Workshop on Syntactic Analysis of Non-Canonical Language*.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2008. The Berkeley Parser. <https://code.google.com/p/berkeleyparser/>.
- Owen Rambow and Aravind Joshi. 1997. A formal look at dependency grammars and phrase-structure grammars, with special consideration of word-order phenomena. In L. Wanner, editor, *Recent Trends in Meaning-Text Theory*, pages 167–190. John Benjamins, Amsterdam and Philadelphia.
- Satoshi Sekine and Michael Collins. 2008. Evalb. <http://nlp.cs.nyu.edu/evalb/>.
- Libin Shen and Aravind Joshi. 2008. LTAG dependency parsing with bidirectional incremental construction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 495–504, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Libin Shen, Lucas Champollion, and Aravind Joshi. 2008. LTAG-spinal and the Treebank: A new resource for incremental, dependency and semantic parsing. *Language Resources and Evaluation*, 42(1):1–19.
- Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2011. OntoNotes 4.0. Linguistic Data Consortium LDC2011T03.

Learning Polylingual Topic Models from Code-Switched Social Media Documents

Nanyun Peng Yiming Wang Mark Dredze
Human Language Technology Center of Excellence
Center for Language and Speech Processing
Johns Hopkins University, Baltimore, MD USA
{npeng1, freewym, mdredze}@jhu.edu

Abstract

Code-switched documents are common in social media, providing evidence for polylingual topic models to infer aligned topics across languages. We present Code-Switched LDA (csLDA), which infers language specific topic distributions based on code-switched documents to facilitate multi-lingual corpus analysis. We experiment on two code-switching corpora (English-Spanish Twitter data and English-Chinese Weibo data) and show that csLDA improves perplexity over LDA, and learns semantically coherent aligned topics as judged by human annotators.

1 Introduction

Topic models (Blei et al., 2003) have become standard tools for analyzing document collections, and topic analyses are quite common for social media (Paul and Dredze, 2011; Zhao et al., 2011; Hong and Davison, 2010; Ramage et al., 2010; Eisenstein et al., 2010). Their popularity owes in part to their data driven nature, allowing them to adapt to new corpora and languages. In social media especially, there is a large diversity in terms of both the topic and language, necessitating the modeling of multiple languages simultaneously. A good candidate for multi-lingual topic analyses are polylingual topic models (Mimno et al., 2009), which learn topics for multiple languages, creating tuples of language specific distributions over monolingual vocabularies for each topic. Polylingual topic models enable cross language analysis by grouping documents by topic regardless of language.

Training of polylingual topic models requires parallel or comparable corpora: document tuples from multiple languages that discuss the same topic. While additional non-aligned documents

```
User 1: ¡Don Samuel es un crack! #VamosMéxico #DaleTri  
RT @User4: Arriba! Viva Mexico! Advanced to GOLD.  
medal match in "Football"!  
User 2: @user1 rodo que tal el nuevo Mountain ?  
User 3: @User1 @User4 wow this is something !! Ja ja ja  
Football well said
```

Figure 1: Three users discuss Mexico’s football team advancing to the Gold medal game in the 2012 Olympics in code-switched Spanish and English.

can be folded in during training, the “glue” documents are required to aid in the alignment across languages. However, the ever changing vocabulary and topics of social media (Eisenstein, 2013) make finding suitable comparable corpora difficult. Standard techniques – such as relying on machine translation parallel corpora or comparable documents extracted from Wikipedia in different languages – fail to capture the specific terminology of social media. Alternate methods that rely on bilingual lexicons (Jagarlamudi and Daumé, 2010) similarly fail to adapt to shifting vocabularies. The result: an inability to train polylingual models on social media.

In this paper, we offer a solution: utilize code-switched social media to discover correlations across languages. Social media is filled with examples of code-switching, where users switch between two or more languages, both in a conversation and even a single message (Ling et al., 2013). This mixture of languages in the same context suggests alignments between words across languages through the common topics discussed in the context.

We learn from code-switched social media by extending the polylingual topic model framework to infer the language of each token and then automatically processing the learned topics to identify aligned topics. Our model improves both in terms of perplexity and a human evaluation, and we provide some example analyses of social media that rely on our learned topics.

2 Code-Switching

Code-switched documents has received considerable attention in the NLP community. Several tasks have focused on identification and analysis, including mining translations in code-switched documents (Ling et al., 2013), predicting code-switched points (Solorio and Liu, 2008a), identifying code-switched tokens (Lignos and Marcus, 2013; Yu et al., 2012; Elfardy and Diab, 2012), adding code-switched support to language models (Li and Fung, 2012), linguistic processing of code switched data (Solorio and Liu, 2008b), corpus creation (Li et al., 2012; Diab and Kamboj, 2011), and computational linguistic analyses and theories of code-switching (Sankoff, 1998; Joshi, 1982).

Code-switching specifically in social media has also received some recent attention. Lignos and Marcus (2013) trained a supervised token level language identification system for Spanish and English code-switched social media to study code-switching behaviors. Ling et al. (2013) mined translation spans for Chinese and English in code-switched documents to improve a translation system, relying on an existing translation model to aid in the identification and extraction task. In contrast to this work, we take an unsupervised approach, relying only on readily available document level language ID systems to utilize code-switched data. Additionally, our focus is not on individual messages, rather we aim to train a model that can be used to analyze entire corpora.

In this work we consider two types of code-switched documents: single messages and conversations, and two language pairs: Chinese-English and Spanish-English. Figure 1 shows an example of a code-switched Spanish-English *conversation*, in which three users discuss Mexico’s football team advancing to the Gold medal game in the 2012 Summer Olympics. In this conversation, some tweets are code-switched and some are in a single language. By collecting the entire conversation into a single document we provide the topic model with additional content. An example of a Chinese-English code-switched messages is given by Ling et al. (2013):

watup Kenny Mayne!! - Kenny Mayne
最近怎么样啊!!

Here a user switches between languages in a single *message*. We empirically evaluate our model on

both conversations and messages. In the model presentation we will refer to both as “documents.”

3 csLDA

To train a polylingual topic model on social media, we make two modifications to the model of Mimno et al. (2009): add a token specific language variable, and a process for identifying aligned topics.

First, polylingual topic models require parallel or comparable corpora in which each document has an assigned language. In the case of code-switched social media data, we require a *per-token* language variable. However, while document level language identification (LID) systems are common place, very few languages have per-token LID systems (King and Abney, 2013; Lignos and Marcus, 2013).

To address the lack of available LID systems, we add a per-token latent language variable to the polylingual topic model. For documents that are not code-switched, we observe these variables to be the output of a document level LID system. In the case of code-switched documents, these variables are inferred during model inference.

Second, polylingual topic models assume the aligned topics are from parallel or comparable corpora, which implicitly assumes that a topics popularity is balanced across languages. Topics that show up in one language necessarily show up in another. However, in the case of social media, we can make no such assumption. The topics discussed are influenced by users, time, and location, all factors intertwined with choice of language. For example, English speakers will more likely discuss Olympic basketball while Spanish speakers football. There may be little or no documents on a given topic in one language, while they are plentiful in another. In this case, a polylingual topic model, which necessarily infers a topic-specific word distribution for each topic in each language, would learn two unrelated word distributions in two languages for a single topic. Therefore, naively using the produced topics as “aligned” across languages is ill-advised.

Our solution is to automatically identify aligned polylingual topics after learning by examining a topic’s distribution across code-switched documents. Our metric relies on distributional properties of an inferred topic across the entire collection.

To summarize, based on the model of Mimno et al. (2009) we will learn:

- For each topic, a language specific word distribution.
- For each (code-switched) token, a language.
- For each topic, an identification as to whether the topic captures an alignment across languages.

The first two goals are achieved by incorporating new hidden variables in the traditional polylingual topic model. The third goal requires an automated post-processing step. We call the resulting model Code-Switched LDA (csLDA). The generative process is as follows:

- For each topic $z \in \mathcal{T}$
 - For each language $l \in \mathcal{L}$
 - Draw word distribution $\phi_z^l \sim \text{Dir}(\beta^l)$
- For each document $d \in \mathcal{D}$:
 - Draw a topic distribution $\theta_d \sim \text{Dir}(\alpha)$
 - Draw a language distribution $\psi_d \sim \text{Dir}(\gamma)$
 - For each token $i \in d$:
 - Draw a topic $z_i \sim \theta_d$
 - Draw a language $l_i \sim \psi_d$
 - Draw a word $w_i \sim \phi_{z_i}^{l_i}$

For monolingual documents, we fix l_i to the LID tag for all tokens. Additionally, we use a single background distribution for each language to capture stopwords; a control variable π , which follows a Dirichlet distribution with prior parameterized by δ , is introduced to decide the choice between background words and topic words following (Chemudugunta et al., 2006)¹. We use asymmetric Dirichlet priors (Wallach et al., 2009), and let the optimization process learn the hyperparameters. The graphical model is shown in Figure 2.

3.1 Inference

Inference for csLDA follows directly from LDA. A Gibbs sampler learns the word distributions ϕ_z^l for each language and topic. We use a block Gibbs sampler to jointly sample topic and language variables for each token. As is customary, we collapse out ϕ , θ and ψ . The sampling posterior is:

$$P(z_i, l_i | \mathbf{w}, \mathbf{z}_{-i}, \mathbf{l}_{-i}, \alpha, \beta, \gamma) \propto \frac{(n_{w_i}^{l_i, z_i})_{-i} + \beta}{n_{-i}^{l_i, z_i} + \mathcal{W}\beta} \times \frac{m_{-i}^{z_i, d} + \alpha}{m_{-i}^d + \mathcal{T}\alpha} \times \frac{o_{-i}^{l_i, d} + \gamma}{o_{-i}^d + \mathcal{L}\gamma} \quad (1)$$

where $(n_{w_i}^{l_i, z_i})_{-i}$ is the number of times the type for word w_i assigned to topic z and language l (ex-

¹Omitted from the generative process but shown in Fig. 2.

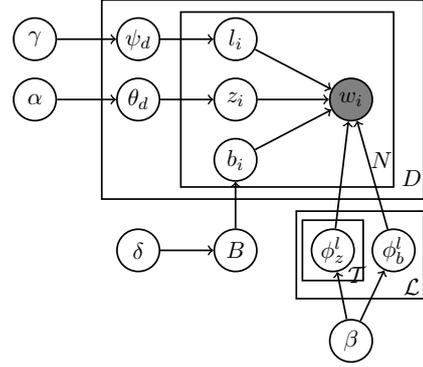


Figure 2: The graphical model for csLDA.

cluding current word w_i), $m_{-i}^{z,d}$ is the number of tokens assigned to topic z in document d (excluding current word w_i), $o_{-i}^{l,d}$ is the number of tokens assigned to language l in document d (excluding current word w_i), and these variables with superscripts or subscripts omitted are totals across all values for the variable. \mathcal{W} is the number of words in the corpus. All counts omit words assigned to the background. During sampling, words are first assigned to the background/topic distribution and then topic and language are sampled for non-background words.

We optimize the hyperparameters α , β , γ and δ by interleaving sampling iterations with a Newton-Raphson update to obtain the MLE estimate for the hyperparameters. Taking α as an example, one step of the Newton-Raphson update is:

$$\alpha^{new} = \alpha^{old} - \mathbf{H}^{-1} \frac{\partial \mathcal{L}}{\partial \alpha} \quad (2)$$

where \mathbf{H} is the Hessian matrix and $\frac{\partial \mathcal{L}}{\partial \alpha}$ is the gradient of the likelihood function with respect to the optimizing hyperparameter. We interleave 200 sampling iterations with one Newton-Raphson update.

3.2 Selecting Aligned Topics

We next identify learned topics (a set of related word-distributions) that truly represent an aligned topic across languages, as opposed to an unrelated set of distributions for which there is no supporting alignment evidence in the corpus. We begin by measuring how often each topic occurs in code-switched documents. If a topic never occurs in a code-switched document, then there can be no evidence to support alignment across languages. For the topics that appear at least once in a code-switched document, we estimate their probability

in the code-switched documents by a MAP estimate of θ . Topics appearing in at least one code-switched document with probability greater than a threshold p are selected as candidates for true cross-language topics.

4 Data

We used two datasets: a Sina Weibo Chinese-English corpus (Ling et al., 2013) and a Spanish-English Twitter corpus.

Weibo Ling et al. (2013) extracted over 1m Chinese-English parallel segments from Sina Weibo, which are code-switched messages. We randomly sampled 29,705 code-switched messages along with 42,116 Chinese and 42,116 English messages from the the same time frame. We used these data for training. We then sampled an additional 2475 code-switched messages, 4221 English and 4211 Chinese messages as test data.

Olympics We collected tweets from July 27, 2012 to August 12, 2012, and identified 302,775 tweets about the Olympics based on related hashtags and keywords (e.g. olympics, #london2012, etc.) We identified code-switched tweets using the Chromium Language Detector². This system provides the top three possible languages for a given document with confidence scores; we identify a tweet as code-switched if two predicted languages each have confidence greater than 33%. We then used the tagger of Lignos and Marcus (2013) to obtain token level LID tags, and only tweets with tokens in both Spanish and English are used as code-switched tweets. In total we identified 822 Spanish-English code-switched tweets. We further expanded the mined tweets to full conversations, yielding 1055 Spanish-English code-switched documents (including both tweets and conversations), along with 4007 English and 4421 Spanish tweets composes our data set. We reserve 10% of the data for testing.

5 Experiments

We evaluated csLDA on the two datasets and evaluated each model using perplexity on held out data and human judgements. While our goal is to learn polylingual topics, we cannot compare to previous polylingual models since they require comparable data, which we lack. Instead, we constructed a baseline from LDA run on the entire dataset (no

language information.) For each model, we measured the document completion perplexity (Rosen-Zvi et al., 2004) on the held out data. We experimented with different numbers of topics (\mathcal{T}). Since csLDA duplicates topic distributions ($\mathcal{T} \times \mathcal{L}$) we used twice as many topics for LDA.

Figure 3 shows test perplexity for varying \mathcal{T} and perplexity for the best setting of csLDA ($\mathcal{T}=60$) and LDA ($\mathcal{T}=120$). The table lists both monolingual and code-switched test data; csLDA improves over LDA in almost every case, and across all values of \mathcal{T} . The background distribution (-bg) has mixed results for LDA, whereas for csLDA it shows consistent improvement. Table 4 shows some csLDA topics. While there are some mistakes, overall the topics are coherent and aligned.

We use the available per-token LID system (Lignos and Marcus, 2013) for Spanish/English to justify csLDA’s ability to infer the hidden language variables. We ran csLDA-bg with l_i set to the value provided by the LID system for code-switched documents (csLDA-bg with LID), which gives csLDA high quality LID labels. While we see gains for the code-switched data, overall the results for csLDA-bg and csLDA-bg with LID are similar, suggesting that the model can operate effectively even without a supervised per-token LID system.

5.1 Human Evaluation

We evaluate topic alignment quality through a human judgements (Chang et al., 2009). For each aligned topic, we show an annotator the 20 most frequent words from the foreign language topic (Chinese or Spanish) with the 20 most frequent words from the aligned English topic and two random English topics. The annotators are asked to select the most related English topic among the three; the one with the most votes is considered the aligned topic. We count how often the model’s alignments agree.

LDA may learn comparable topics in different languages but gives no explicit alignments. We create alignments by classifying each LDA topic by language using the KL-divergence between the topic’s words distribution and a word distribution for the English/foreign language inferred from the monolingual documents. Language is assigned to a topic by taking the minimum KL. For Weibo data, this was not effective since the vocabularies of each language are highly unbalanced. Instead,

²<https://code.google.com/p/chromium-compact-language-detector/>

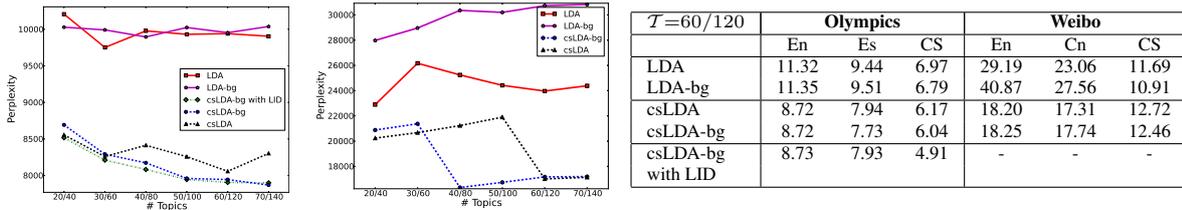


Figure 3: Plots show perplexity for different \mathcal{T} (Olympics left, Weibo right). Perplexity in the table are in magnitude of 1×10^3 .

Football		Basketball		Social Media		Transportation	
English	Spanish	English	Spanish	English	Chinese	English	Chinese
mexico	mucho	game	españa	twitter	啊啊啊	car	汽车
brazil	argentina	basketball	baloncesto	bitly	微博	drive	这个
soccer	méxico	year	basketball	facebook	更新	road	真真
vs	brasil	finals	bronze	check	下载	line	明年
womens	ganará	gonna	china	use	转发	train	自行车
football	tri	nba	final	blog	视频	harry	车型
mens	yahel_castillo	obama	rusia	free	pm	汽车	奔驰
final	delpo	lebron	española	post	推特	bus	大众

Figure 4: Examples of aligned topics from Olympics (left) and Weibo (right).

we manually labeled the topics by language. We then pair topics across languages using the cosine similarity of their co-occurrence statistics in code-switched documents. Topic pairs with similarity above t are considered aligned topics. We also used a threshold p (§3.2) to select aligned topics in csLDA. To ensure a fair comparison, we select the same number of aligned topics for LDA and csLDA.³ We used the best performing setting: csLDA $\mathcal{T}=60$, LDA $\mathcal{T}=120$, which produced 12 alignments from Olympics and 28 from Weibo.

Using Mechanical Turk we collected multiple judgements per alignment. For Spanish, we removed workers who disagreed with the majority more than 50% of the time (83 deletions), leaving 6.5 annotations for each alignment (85.47% inter-annotator agreement.) For Chinese, since quality of general Chinese turkers is low (Pavlick et al., 2014) we invited specific workers and obtained 9.3 annotations per alignment (78.72% inter-annotator agreement.) For Olympics, LDA alignments matched the judgements 25% of the time, while csLDA matched 50% of the time. While csLDA found 12 alignments and LDA 29, the 12 topics evaluated from both models show that csLDA’s alignments are higher quality. For the Weibo data, LDA matched judgements 71.4%, while csLDA matched 75%. Both obtained high

³We used thresholds $p = 0.2$ and $t = 0.0001$. We limited the model with more alignments to match the one with less.

quality alignments – likely due both to the fact that the code-switched data is curated to find translations and we hand labeled topic language – but csLDA found many more alignments: 60 as compared to 28. These results confirm our automated results: csLDA finds higher quality topics that span both languages.

References

- David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research (JMLR)*, 3:993–1022.
- Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L Boyd-graber, and David M Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Advances in neural information processing systems*, pages 288–296.
- Chaitanya Chemudugunta, Padhraic Smyth, and Mark Steyvers. 2006. Modeling general and specific aspects of documents with a probabilistic topic model. In *NIPS*.
- Mona Diab and Ankit Kamboj. 2011. Feasibility of leveraging crowd sourcing for the creation of a large scale annotated resource for Hindi English code switched data: A pilot annotation. In *Proceedings of the 9th Workshop on Asian Language Resources*, pages 36–40, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.
- 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*, pages 1277–1287. Association for Computational Linguistics.
- Jacob Eisenstein. 2013. What to do about bad language on the internet. In *NAACL*.
- Heba Elfardy and Mona Diab. 2012. Token level identification of linguistic code switching. In *Proceedings of COLING 2012: Posters*, pages 287–296, Mumbai, India, December. The COLING 2012 Organizing Committee.
- Liangjie Hong and Brian D Davison. 2010. Empirical study of topic modeling in twitter. In *Proceedings of the First Workshop on Social Media Analytics*, pages 80–88. ACM.
- Jagadeesh Jagarlamudi and Hal Daumé. 2010. Extracting multilingual topics from unaligned comparable corpora. *Advances in Information Retrieval*, pages 444–456.
- Aravind K Joshi. 1982. Processing of sentences with intra-sentential code-switching. In *Proceedings of the 9th Conference on Computational linguistics (COLING)*, pages 145–150.
- Ben King and Steven Abney. 2013. Labeling the languages of words in mixed-language documents using weakly supervised methods. In *NAACL*.
- Ying Li and Pascale Fung. 2012. Code-switch language model with inversion constraints for mixed language speech recognition. In *Proceedings of COLING 2012*, pages 1671–1680, Mumbai, India, December. The COLING 2012 Organizing Committee.
- Ying Li, Yue Yu, and Pascale Fung. 2012. A mandarin-english code-switching corpus. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 2515–2519, Istanbul, Turkey, May. European Language Resources Association (ELRA). ACL Anthology Identifier: L12-1573.
- Constantine Lignos and Mitch Marcus. 2013. Toward web-scale analysis of codeswitching. In *Annual Meeting of the Linguistic Society of America*.
- Wang Ling, Guang Xiang, Chris Dyer, Alan Black, and Isabel Trancoso. 2013. Microblogs as parallel corpora. In *Proceedings of the 51st Annual Meeting on Association for Computational Linguistics, ACL '13*. Association for Computational Linguistics.
- David Mimno, Hanna M Wallach, Jason Naradowsky, David A Smith, and Andrew McCallum. 2009. Polylingual topic models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 880–889. Association for Computational Linguistics.
- Michael J Paul and Mark Dredze. 2011. You are what you tweet: Analyzing twitter for public health. In *ICWSM*.
- Ellie Pavlick, Matt Post, Ann Irvine, Dmitry Kachaev, and Chris Callison-Burch. 2014. The language demographics of Amazon Mechanical Turk. *Transactions of the Association for Computational Linguistics*, 2(Feb):79–92.
- Daniel Ramage, Susan T Dumais, and Daniel J Liebling. 2010. Characterizing microblogs with topic models. In *ICWSM*.
- Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. 2004. The author-topic model for authors and documents. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494. AUAI Press.
- David Sankoff. 1998. The production of code-mixed discourse. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 8–21, Montreal, Quebec, Canada, August. Association for Computational Linguistics.
- Tamar Solorio and Yang Liu. 2008a. Learning to predict code-switching points. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 973–981, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Tamar Solorio and Yang Liu. 2008b. Part-of-Speech tagging for English-Spanish code-switched text. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1060, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Hanna M Wallach, David M Mimno, and Andrew McCallum. 2009. Rethinking lda: Why priors matter. In *NIPS*, volume 22, pages 1973–1981.
- Liang-Chih Yu, Wei-Cheng He, and Wei-Nan Chien. 2012. A language modeling approach to identifying code-switched sentences and words. In *Proceedings of the Second CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 3–8, Tianjin, China, December. Association for Computational Linguistics.
- Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. 2011. Comparing twitter and traditional media using topic models. In *Advances in Information Retrieval*, pages 338–349. Springer.

Normalizing tweets with edit scripts and recurrent neural embeddings

Grzegorz Chrupała

Tilburg Center for Cognition and Communication

Tilburg University

g.chrupala@uvt.nl

Abstract

Tweets often contain a large proportion of abbreviations, alternative spellings, novel words and other non-canonical language. These features are problematic for standard language analysis tools and it can be desirable to convert them to canonical form. We propose a novel text normalization model based on learning edit operations from labeled data while incorporating features induced from unlabeled data via character-level neural text embeddings. The text embeddings are generated using an Simple Recurrent Network. We find that enriching the feature set with text embeddings substantially lowers word error rates on an English tweet normalization dataset. Our model improves on state-of-the-art with little training data and without any lexical resources.

1 Introduction

A stream of posts from Twitter contains text written in a large variety of languages and writing systems, in registers ranging from formal to internet slang. Substantial effort has been expended in recent years to adapt standard NLP processing pipelines to be able to deal with such content. One approach has been text normalization, i.e. transforming tweet text into a more canonical form which standard NLP tools expect. A multitude of resources and approaches have been used to deal with normalization: hand-crafted and (semi-)automatically induced dictionaries, language models, finite state transducers, machine translation models and combinations thereof. Methods such as those of Han and Baldwin (2011), Liu et al. (2011), Gouws et al. (2011) or Han et al. (2012) are unsupervised but they typically use many adjustable parameters which

need to be tuned on some annotated data. In this work we suggest a simple, supervised character-level string transduction model which easily incorporates features automatically learned from large amounts of unlabeled data and needs only a limited amount of labeled training data and no lexical resources.

Our model learns sequences of edit operations from labeled data using a Conditional Random Field (Lafferty et al., 2001). Unlabeled data is incorporated following recent work on using character-level text embeddings for text segmentation (Chrupała, 2013), and word and sentence boundary detection (Evang et al., 2013). We train a recurrent neural network language model (Mikolov et al., 2010; Mikolov, 2012b) on a large collection of tweets. When run on new strings, the activations of the units in the hidden layer at each position in the string are recorded and used as features for training the string transduction model.

The principal contributions of our work are: (i) we show that a discriminative sequence labeling model is apt for text normalization and performs at state-of-the-art levels with small amounts of labeled training data; (ii) we show that character-level neural text embeddings can be used to effectively incorporate information from unlabeled data into the model and can substantially boost text normalization performance.

2 Methods

Many approaches to text normalization adopt the noisy channel setting, where the model normalizing source string s into target canonical form t is factored into two parts: $\hat{t} = \arg \max_t P(t)P(s|t)$. The error term $P(s|t)$ models how canonical strings are transformed into variants such as e.g. misspellings, emphatic lengthenings or abbreviations. The language model $P(t)$ encodes which target strings are probable.

We think this decomposition is less appropriate

Input	c	␣	w	a	t
Edit	DEL	INS(see)	NIL	INS(h)	NIL
Output		see␣	w	ha	t

Table 1: Example edit script.

in the context of text normalization than in applications from which it was borrowed such as Machine Translations. This is because it is not obvious what kind of data can be used to estimate the language model: there is plentiful text from the source domain, but little of it is in normalized *target* form. There is also much edited text such as news text, but it comes from a very different domain. One of the main advantages of the noisy channel decomposition is that it makes it easy to exploit large amounts of unlabeled data in the form of a language model. This advantage does not hold for text normalization.

We thus propose an alternative approach where normalization is modeled directly, and which enables easy incorporation of unlabeled data from the *source* domain.

2.1 Learning to transduce strings

Our string transduction model works by learning the sequence of edits which transform the input string into the output string. Given a pair of strings such a sequence of edits (known as the shortest edit script) can be found using the DIFF algorithm (Miller and Myers, 1985; Myers, 1986). Our version of DIFF uses the following types of edits:

- NIL – no edits,
- DEL – delete character at this position,
- INS(\cdot) – insert specified string before character at this position.¹

Table 1 shows a shortest edit script for the pair of strings (*c wat, see what*).

We use a sequence labeling model to learn to label input strings with edit scripts. The training data for the model is generated by computing shortest edit scripts for pairs of original and normalized strings. As a sequence labeler we use Conditional Random Fields (Lafferty et al., 2001). Once trained the model is used to label new strings and the predicted edit script is applied to the input string producing the normalized output string. Given source string s the predicted target string \hat{t}

¹The input string is extended with an empty symbol to account for the cases where an insertion is needed at the end of the string.

is:

$$\hat{t} = \arg \max_t P(\text{ses}(s, t) | s)$$

where $e = \text{ses}(s, t)$ is the shortest edit script mapping s to t . $P(e|s)$ is modeled with a linear-chain Conditional Random Field.

2.2 Character-level text embeddings

Simple Recurrent Networks (SRNs) were introduced by Elman (1990) as models of temporal, or sequential, structure in data, including linguistic data (Elman, 1991). More recently SRNs were used as language models for speech recognition and shown to outperform classical n-gram language models (Mikolov et al., 2010; Mikolov, 2012b). Another version of recurrent neural nets has been used to generate plausible text with a character-level language model (Sutskever et al., 2011). We use SRNs to induce character-level text representations from unlabeled Twitter data to use as features in the string transduction model.

The units in the hidden layer at time t receive connections from input units at time t and also from the hidden units at the previous time step $t - 1$. The hidden layer predicts the state of the output units at the next time step $t + 1$. The input vector $w(t)$ represents the input element at current time step, here the current character. The output vector $y(t)$ represents the predicted probabilities for the next character. The activation s_j of a hidden unit j is a function of the current input and the state of the hidden layer at the previous time step: $t - 1$:

$$s_j(t) = \sigma \left(\sum_{i=1}^I w_i(t) U_{ji} + \sum_{l=1}^L s_j(t-1) W_{jl} \right)$$

where σ is the sigmoid function and U_{ji} is the weight between input component i and hidden unit j , while W_{jl} is the weight between hidden unit l at time $t - 1$ and hidden unit j at time t . The representation of recent history is stored in a limited number of recurrently connected hidden units. This forces the network to make the representation compressed and abstract rather than just memorize literal history. Chrupała (2013) and Evang et al. (2013) show that these text embeddings can be useful as features in textual segmentation tasks. We use them to bring in information from unlabeled data into our string transduction model and then train a character-level SRN language model on unlabeled tweets. We run the trained model on

```

@YuszLAL100A 暇すぎるwwwwwとか雑役者についてる... (> >
晒せ 信じに行けていいんだな... RT @yaepdrrafa:
@fsch_chany siaaa,, dobek taha subus sama kiri kabur
wanak... hahah
なかなかない。
やばい
But I'm the good first-Good Chulc

```

Figure 1: Tweets randomly generated with an SRN

new tweets and record the activation of the hidden layer at each position as the model predicts the next character. These activation vectors form our text embeddings: they are discretized and used as input features to the supervised sequence labeler as described in Section 3.4.

3 Experimental Setup

We limit the size of the string alphabet by always working with UTF-8 encoded strings, and using bytes rather than characters as basic units.

3.1 Unlabeled tweets

In order to train our SRN language model we collected a set of tweets using the Twitter sampling API. We use the raw sample directly without filtering it in any way, relying on the SRN to learn the structure of the data. The sample consists of 414 million bytes of UTF-8 encoded in a variety of languages and scripts text. We trained a 400-hidden-unit SRN, to predict the next byte in the sequence using backpropagation through time. Input bytes were encoded using one-hot representation. We modified the RNNLM toolkit (Mikolov, 2012a) to record the activations of the hidden layer and ran it with the default learning rate schedule. Given that training SRNs on large amounts of text takes a considerable amount of time we did not vary the size of the hidden layer. We did try to filter tweets by language and create specific embeddings for English but this had negligible effect on tweet normalization performance.

The trained SRN language model can be used to generate random text by sampling the next byte from its predictive distribution and extending the string with the result. Figure 1 shows example strings generated in this way: the network seems to prefer to output pseudo-tweets written consistently in a single script with words and pseudo-words mostly from a single language. The generated byte sequences are valid UTF-8 strings.

In Table 2 in the first column we show the suffix of a string for which the SRN is predicting the last byte. The rest of each row shows the nearest neighbors of this string in embedding space, i.e.

should h	should d	will s	will m	should a
@justth	@neenu	@raven_	@lanae	@despic
maybe	u maybe y	cause i	wen i	when i

Table 2: Nearest neighbors in embedding space.

strings for which the SRN is activated in a similar way when predicting its last byte as measured by cosine similarity.

3.2 Normalization datasets

A difficulty in comparing approaches to tweet normalization is the sparsity of publicly available datasets. Many authors evaluate on private tweet collections and/or on the text message corpus of Choudhury et al. (2007).

For English, Han and Baldwin (2011) created a small tweet dataset annotated with normalized variants at the word level. It is hard to interpret the results from Han and Baldwin (2011), as the evaluation is carried out by assuming that the words to be normalized are known in advance: Han et al. (2012) remedy this shortcoming by evaluating a number of systems without pre-specifying ill-formed tokens. Another limitation is that only word-level normalization is covered in the annotation; e.g. splitting or merging of words is not allowed. The dataset is also rather small: 549 tweets, which contain 2139 annotated out-of-vocabulary (OOV) words. Nevertheless, we use it here for training and evaluating our model. This dataset does not specify a development/test split. In order to maximize the size of the training data while avoiding tuning on test data we use a split cross-validation setup: we generate 10 cross-validation folds, and use 5 of them during development to evaluate variants of our model. The best performing configuration is then evaluated on the remaining 5 cross-validation folds.

3.3 Model versions

The simplest way to normalize tweets with a string transduction model is to treat whole tweets as input sequences. Many other tweet normalization methods work in a word-wise fashion: they first identify OOV words and then replace them with normalized forms. Consequently, publicly available normalization datasets are annotated at word level. We can emulate this setup by training the sequence labeler on words, instead of whole tweets. This approach sacrifices some generality, since transformations involving multiple words cannot

be learned. However, word-wise models are more comparable with previous work. We investigated the following models:

- OOV-ONLY is trained on individual words and in-vocabulary (IV) words are discarded for training, and left unchanged for prediction.²
- ALL-WORDS is trained on all words and allowed to change IV words.
- DOCUMENT is trained on whole tweets.

Model OOV-ONLY exploits the setting when the task is constrained to only normalize words absent from a reference dictionary, while DOCUMENT is the one most generally applicable but does not benefit from any constraints. To keep model size within manageable limits we reduced the label set for models ALL-WORDS and DOCUMENT by replacing labels which occur less than twice in the training data with NIL. For OOV-ONLY we were able to use the full label set. As our sequence labeling model we use the Wapiti implementation of Conditional Random Fields (Lavergne et al., 2010) with the L-BFGS optimizer and elastic net regularization with default settings.

3.4 Features

We run experiments with two feature sets: N-GRAM and N-GRAM+SRN. N-GRAM are character n-grams of size 1–3 in a window of $(-2, +2)$ around the current position. For the N-GRAM+SRN feature set we augment N-GRAM with features derived from the activations of the hidden units as the SRN is trying to predict the current character. In order to use the activations in the CRF model we discretize them as follows. For each of the $K = 10$ most active units out of total $J = 400$ hidden units, we create features $(f(1) \dots f(K))$ defined as $f(k) = 1$ if $s_{j(k)} > 0.5$ and $f(k) = 0$ otherwise, where $s_{j(k)}$ returns the activation of the k^{th} most active unit.

3.5 Evaluation metrics

As our evaluation metric we use word error rate (WER) which is defined as the Levenshtein edit distance between the predicted word sequence \hat{t} and the target word sequence t , normalized by the total number of words in the target string. A more generally applicable metric would be character error rate, but we report WERs to make our results easily comparable with previous work. Since the

²We used the IV/OOV annotations in the Han et al. (2012) dataset, which are automatically derived from the aspell dictionary.

Model	Features	WER (%)
NO-OP		11.7
DOCUMENT	NGRAM	6.8
DOCUMENT	NGRAM+SRN	5.7
ALL WORDS	NGRAM	7.2
ALL WORDS	NGRAM+SRN	5.0
OOV-ONLY	NGRAM	5.1
OOV-ONLY	NGRAM+SRN	4.5

Table 3: WERs on development data.

9 cont continued	5 gon gonna
4 bro brother	4 congrats congratulations
3 yall you	3 pic picture
2 wuz what’s	2 mins minutes
2 juss just	2 fb facebook

Table 4: Improvements from SRN features.

English dataset is pre-tokenized and only covers word-to-word transformations, this choice has little importance here and character error rates show a similar pattern to word error rates.

4 Results

Table 3 shows the results of our development experiments. NO-OP is a baseline which leaves text unchanged. As expected the most constrained model OOV-ONLY outperforms the more generic models on this dataset. For all model variations, adding SRN features substantially improves performance: the relative error reductions range from 12% for OOV-ONLY to 30% for ALL-WORDS. Table 4 shows the non-unique normalizations made by the OOV-ONLY model with SRN features which were missed without them. SRN features seem to be especially useful for learning long-range, multi-character edits, e.g. *fb* for *facebook*.

Table 5 shows the non-unique normalizations which were missed by the best model: they are a mixture of relatively standard variations which happen to be infrequent in our data, like *tonite* or *gf*, and a few idiosyncratic respellings like *uu* or *bhee*. Our supervised approach makes it easy to address the first type of failure by simply annotating additional training examples.

Table 6 presents evaluation results of several approaches reported in Han et al. (2012) as well as the model which did best in our development experiments. HB-dict is the Internet slang dictionary from Han and Baldwin (2011). GHM-dict is the automatically constructed dictionary from

4 1 one	2 withh with
2 uu you	2 tonite tonight
2 thx thanks	2 thiis this
2 smh somehow	2 outta out
2 n in	2 m am
2 hmwrk homework	2 gf girlfriend
2 fxckin fucking	2 dha the
2 de the	2 d the
2 bhee be	2 bb baby

Table 5: Missed transformations.

Method	WER (%)
NO-OP	11.2
HB-dict	6.6
GHM-dict	7.6
S-dict	9.7
Dict-combo	4.9
Dict-combo+HB-norm	7.9
OOV-ONLY NGRAM+SRN (test)	4.8

Table 6: WERs compared to previous work.

Gouws et al. (2011); S-dict is the automatically constructed dictionary from (Han et al., 2012); Dict-combo are all the dictionaries combined and Dict-combo+HB-norm are all dictionaries combined with approach of Han and Baldwin (2011). The WER reported for OOV-ONLY NGRAM+SRN is on the test folds only. The score on the full dataset is a bit better: 4.66%. As can be seen our approach it the best performing approach overall and in particular it does much better than all of the single dictionary-based methods. Only the combination of all the dictionaries comes close in performance.

5 Related work

In the field of tweet normalization the approach of Liu et al. (2011, 2012) shows some similarities to ours: they gather a collection of OOV words together with their canonical forms from the web and train a character-level CRF sequence labeler on the edit sequences computed from these pairs. They use this as the error model in a noisy-channel setup combined with a unigram language model. In addition to character n-gram features they use phoneme and syllable features, while we rely on the SRN embeddings to provide generalized representations of input strings.

Kaufmann and Kalita (2010) trained a phrase-based statistical translation model on a parallel

text message corpus and applied it to tweet normalization. In comparison to our first-order linear-chain CRF, an MT model with reordering is more flexible but for this reason needs more training data. It also suffers from language model mismatch mentioned in Section 2: optimal results were obtained by using a low weight for the language model trained on a balanced text corpus.

Many other approaches to tweet normalization are more unsupervised in nature (e.g. Han and Baldwin, 2011; Gouws et al., 2011; Xue et al., 2011; Han et al., 2012). They still require annotated development data for tuning parameters and a variety of heuristics. Our approach works well with similar-sized training data, and unlike unsupervised approaches can easily benefit from more if it becomes available. Further afield, our work has connections to research on morphological analysis: for example Chrupała et al. (2008) use edit scripts to learn lemmatization rules while Dreyer et al. (2008) propose a discriminative model for string transductions and apply it to morphological tasks. While Chrupała (2013) and Evang et al. (2013) use character-level SRN text embeddings for learning segmentation, and recurrent nets themselves have been used for sequence transduction (Graves, 2012), to our knowledge *neural text embeddings* have not been previously applied to string transduction.

6 Conclusion

Learning sequences of edit operations from examples while incorporating unlabeled data via neural text embeddings constitutes a compelling approach to tweet normalization. Our results are especially interesting considering that we trained on only a small annotated data set and did not use any other manually created resources such as dictionaries. We want to push performance further by expanding the training data and incorporating existing lexical resources. It will also be important to check how our method generalizes to other language and datasets (e.g. de Clercq et al., 2013; Alegria et al., 2013).

The general form of our model can be used in settings where normalization is not limited to word-to-word transformations. We are planning to find or create data with such characteristics and evaluate our approach under these conditions.

References

- Iñaki Alegria, Nora Aranberri, Víctor Fresno, Pablo Gamallo, Lluís Padró, Iñaki San Vicente, Jordi Turmo, and Arkaitz Zubiaga. 2013. Introducción a la tarea compartida Tweet-Norm 2013: Normalización léxica de tuits en español. In *Workshop on Tweet Normalization at SEPLN (Tweet-Norm)*, pages 36–45.
- 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 (IJ DAR)*, 10(3-4):157–174.
- Grzegorz Chrupała. 2013. Text segmentation with character-level text embeddings. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*.
- Grzegorz Chrupała, Georgiana Dinu, and Josef Van Genabith. 2008. Learning morphology with Morfette. In *Proceedings of the 6th edition of the Language Resources and Evaluation Conference*.
- Orphée de Clercq, Bart Desmet, Sarah Schulz, Els Lefever, and Véronique Hoste. 2013. Normalization of Dutch user-generated content. In *9th International Conference on Recent Advances in Natural Language Processing (RANLP-2013)*, pages 179–188. INCOMA Ltd.
- Markus Dreyer, Jason R Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1080–1089. Association for Computational Linguistics.
- Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.
- Jeffrey L Elman. 1991. Distributed representations, simple recurrent networks, and grammatical structure. *Machine learning*, 7(2-3):195–225.
- Kilian Evang, Valerio Basile, Grzegorz Chrupała, and Johan Bos. 2013. Elephant: Sequence labeling for word and sentence segmentation. In *Empirical Methods in Natural Language Processing*.
- Stephan Gouws, Dirk Hovy, and Donald Metzler. 2011. Unsupervised mining of lexical variants from noisy text. In *Proceedings of the First workshop on Unsupervised Learning in NLP*, pages 82–90. Association for Computational Linguistics.
- Alex Graves. 2012. Sequence transduction with recurrent neural networks. arXiv:1211.3711.
- Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a# twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 368–378. Association for Computational Linguistics.
- Bo Han, Paul Cook, and Timothy Baldwin. 2012. Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 421–432. Association for Computational Linguistics.
- Max Kaufmann and Jugal Kalita. 2010. Syntactic normalization of Twitter messages. In *International conference on natural language processing, Kharagpur, India*.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01*, pages 282–289. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 504–513. Association for Computational Linguistics.
- Fei Liu, Fuliang Weng, and Xiao Jiang. 2012. A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 1035–1044. Association for Computational Linguistics.
- Fei Liu, Fuliang Weng, Bingqing Wang, and Yang Liu. 2011. Insertion, deletion, or substitution?: normalizing text messages without pre-categorization nor supervision. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 71–76. Association for Computational Linguistics.
- Tomáš Mikolov. 2012a. Recurrent neural network language models. <http://rnnlm.org>.
- Tomáš Mikolov. 2012b. *Statistical language models based on neural networks*. Ph.D. thesis, Brno University of Technology.
- Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *InterSpeech*, pages 1045–1048.
- Webb Miller and Eugene W Myers. 1985. A file comparison program. *Software: Practice and Experience*, 15(11):1025–1040.
- Eugene W Myers. 1986. An O(ND) difference algorithm and its variations. *Algorithmica*, 1(1-4):251–266.

Ilya Sutskever, James Martens, and Geoffrey E Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024.

Zhenzhen Xue, Dawei Yin, and Brian D Davison. 2011. Normalizing microtext. In *Proceedings of the AAAI-11 Workshop on Analyzing Microtext*, pages 74–79.

Exponential Reservoir Sampling for Streaming Language Models

Miles Osborne*

School of Informatics
University of Edinburgh

Ashwin Lall

Mathematics and Computer Science
Denison University

Benjamin Van Durme

HLTCOE
Johns Hopkins University

Abstract

We show how rapidly changing textual streams such as Twitter can be modelled in fixed space. Our approach is based upon a randomised algorithm called *Exponential Reservoir Sampling*, unexplored by this community until now. Using language models over Twitter and Newswire as a testbed, our experimental results based on perplexity support the intuition that recently observed data generally outweighs that seen in the past, but that at times, the past can have valuable signals enabling better modelling of the present.

1 Introduction

Work by Talbot and Osborne (2007), Van Durme and Lall (2009) and Goyal et al. (2009) considered the problem of building very large language models via the use of randomized data structures known as *sketches*.¹ While efficient, these structures still scale linearly in the number of items stored, and do not handle deletions well: if processing an unbounded stream of text, with new words and phrases being regularly added to the model, then with a fixed amount of space, errors will only increase over time. This was pointed out by Levenberg and Osborne (2009), who investigated an alternate approach employing perfect-hashing to allow for deletions over time. Their deletion criterion was task-specific and based on how a machine translation system queried a language model.

Corresponding author: miles@inf.ed.ac.uk

¹Sketches provide space efficiencies that are measured on the order of individual bits per item stored, but at the cost of being lossy: sketches trade off space for error, where the less space you use, the more likely you will get erroneous responses to queries.

Here we ask what the appropriate selection criterion is for streaming data based on a non-stationary process, when concerned with an intrinsic measure such as perplexity. Using Twitter and newswire, we pursue this via a sampling strategy: we construct models over sentences based on a sample of previously observed sentences, then measure perplexity of incoming sentences, all on a day by day, rolling basis. Three sampling approaches are considered: A fixed-width sliding window of most recent content, uniformly at random over the stream and a biased sample that prefers recent history over the past.

We show experimentally that a moving window is better than uniform sampling, and further that exponential (biased) sampling is best of all. For streaming data, recently encountered data is valuable, but there is also signal in the previous stream.

Our sampling methods are based on *reservoir sampling* (Vitter, 1985), a popularly known method in some areas of computer science, but which has seen little use within computational linguistics.² Standard reservoir sampling is a method for maintaining a uniform sample over a dynamic stream of elements, using constant space. Novel to this community, we consider a variant owing to Aggarwal (2006) which provides for an exponential bias towards recently observed elements. This *exponential reservoir sampling* has all of the guarantees of standard reservoir sampling, but as we show, is a better fit for streaming textual data. Our approach is fully general and can be applied to any streaming task where we need to model the present and can only use fixed space.

²Exceptions include work by Van Durme and Lall (2011) and Van Durme (2012), aimed at different problems than that explored here.

2 Background

We address two problems: language changes over time, and the observation that space is a problem, even for compact sketches.

Statistical language models often assume either a local Markov property (when working with utterances, or sentences), or that content is generated fully i.i.d. (such as in document-level topic models). However, language shows observable priming effects, sometimes called *triggers*, where the occurrence of a given term decreases the surprisal of some other term later in the same discourse (Lau et al., 1993; Church and Gale, 1995; Beeferman et al., 1997; Church, 2000). Conventional cache and trigger models typically do not deal with new terms and can be seen as adjusting the parameters of a fixed model.

Accounting for previously unseen entries in a language model can be naively simple: as they appear in new training data, add them to the model! However in practice we are constrained by available *space*: how many unique phrases can we store, given the target application environment?

Our work is concerned with modeling language that might change over time, in accordance with current trending discourse topics, but under a strict space constraint. With a fixed amount of memory available, we cannot allow our list of unique words or phrases to grow over time, even while new topics give rise to novel names of people, places, and terms of interest. Thus we need an approach that keeps the size of the model constant, but that is geared to what is being discussed now, as compared to some time in the past.

3 Reservoir Sampling

3.1 Uniform Reservoir Sampling

The reservoir sampling algorithm (Vitter, 1985) is the classic method of sampling without replacement from a stream in a single pass when the length of the stream is of indeterminate or unbounded length. Say that the size of the desired sample is k . The algorithm proceeds by retaining the first k items of the stream and then sampling each subsequent element with probability $f(k, n) = k/n$, where n is the length of the stream so far. (See Algorithm 1.) It is easy to show via induction that, at any time, all the items in the stream so far have equal probability of appearing in the reservoir.

The algorithm processes the stream in a single pass—that is, once it has processed an item in the stream, it does not revisit that item unless it is stored in the reservoir. Given this restriction, the incredible feature of this algorithm is that it is able to guarantee that the samples in the reservoir are a uniformly random sample with no unintended biases even as the stream evolves. This makes it an excellent candidate for situations when the stream is continuously being updated and it is computationally infeasible to store the entire stream or to make more than a single pass over it. Moreover, it is an extremely efficient algorithm as it requires $O(1)$ time (independent of the reservoir size and stream length) for each item in the stream.

Algorithm 1 Reservoir Sampling Algorithm

Parameters:

k : maximum size of reservoir

- 1: Initialize an empty reservoir (any container data type).
 - 2: $n := 1$
 - 3: **for** each item in the stream **do**
 - 4: **if** $n < k$ **then**
 - 5: insert current item into the reservoir
 - 6: **else**
 - 7: with probability $f(n, k)$, eject an element of the reservoir chosen uniformly at random and insert current item into the reservoir
 - 8: $n := n + 1$
-

3.2 Non-uniform Reservoir Sampling

Here we will consider generalizations of the reservoir sampling algorithm in which the sample items in the reservoir are more biased towards the present. Put another way, we will continuously decay the probability that an older item will appear in the reservoir. Models produced using such biases put more modelling stress on the present than models produced using data that is selected uniformly from the stream. The goal here is to continuously update the reservoir sample in such a way that the decay of older items is done consistently while still maintaining the benefits of reservoir sampling, including the single pass and memory/time constraints.

The time-decay scheme we will study in this paper is *exponential bias* towards newer items in the stream. More precisely, we wish for items that

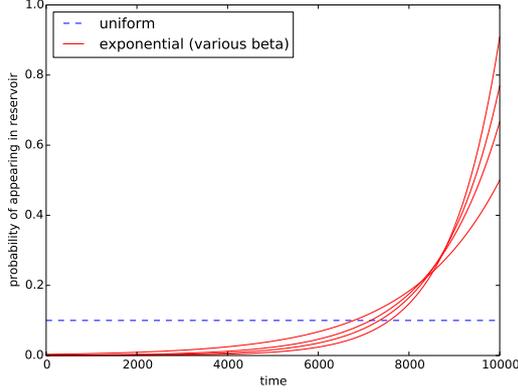


Figure 1: Different biases for sampling a stream

have age a in the stream to appear with probability

$$g(a) = c \cdot \exp(-a/\beta),$$

where a is the age of the item, β is a scale parameter indicating how rapidly older items should be deemphasized, and c is a normalization constant. To give a sense of what these time-decay probabilities look like, some exponential distributions are plotted (along with the uniform distribution) in Figure 1.

Aggarwal (2006) studied this problem and showed that by altering the sampling probability ($f(n, k)$ in Algorithm 1) in the reservoir sampling algorithm, it is possible to achieve different age-related biases in the sample. In particular, he showed that by setting the sampling probability to the constant function $f(n, k) = k/\beta$, it is possible to approximately achieve exponential bias in the sample with scale parameter β (Aggarwal, 2006). Aggarwal’s analysis relies on the parameter β being very large. In the next section we will make the analysis more precise by omitting any such assumption.

3.3 Analysis

In this section we will derive an expression for the bias introduced by an arbitrary sampling function f in Algorithm 1. We will then use this expression to derive the precise sampling function needed to achieve exponential decay.³ Careful selection of f allows us to achieve anything from zero decay (i.e., uniform sampling of the entire stream) to exponential decay. Once again, note that since we are only changing the sampling function, the

³Specifying an arbitrary decay function remains an open problem.

one-pass, memory- and time-efficient properties of reservoir sampling are still being preserved.

In the following analysis, we fix n to be the size of the stream at some fixed time and k to be the size of the reservoir. We assume that the i th element of the stream is sampled with probability $f(i, k)$, for $i \leq n$. We can then derive the probability that an element of age a will still be in the reservoir as

$$g(a) = f(n - a, k) \prod_{t=n-a+1}^n \left(1 - \frac{f(t, k)}{k}\right),$$

since it would have been sampled with probability $f(n - a, k)$ and had independent chances of being replaced at times $t = n - a + 1, \dots, n$ with probability $f(t, k)/k$. For instance, when $f(x, k) = \frac{k}{x}$, the above formula simplifies down to $g(a) = \frac{k}{n}$ (i.e., the uniform sampling case).

For the exponential case, we fix the sampling rate to some constant $f(n, k) = p_k$, and we wish to determine what value to use for p_k to achieve a given exponential decay rate $g(a) = ce^{-a/\beta}$, where c is the normalization constant (to make g a probability distribution) and β is the scale parameter of the exponential distribution. Substituting $f(n, k) = p_k$ in the above formula and equating with the decay rate, we get that $p_k(1 - p_k/k)^a \equiv ce^{-a/\beta}$, which must hold true for all possible values of a . After some algebra, we get that when $f(x, k) = p_k = k(1 - e^{-1/\beta})$, the probability that an item with age a is included in the reservoir is given by the exponential decay rate $g(a) = p_k e^{-a/\beta}$. Note that, for very large values of β , this probability is approximately equal to $p_k \approx k/\beta$ (by using the approximation $e^{-x} \approx 1 - x$, when $|x|$ is close to zero), as given by Aggarwal, but our formula gives the precise sampling probability and works even for smaller values of β .

4 Experiments

Our experiments use two streams of data to illustrate exponential sampling: Twitter and a more conventional newswire stream. The Twitter data is interesting as it is very multilingual, bursty (for example, it talks about memes, breaking news, gossip etc) and written by literally millions of different people. The newswire stream is a lot more well behaved and serves as a control.

4.1 Data, Models and Evaluation

We used one month of chronologically ordered Twitter data and divided it into 31 equal sized

Stream	Interval	Total (toks)	Test (toks)
Twitter	Dec 2013	3282M	105M
Giga	1994 – 2010	635.5M	12M

Table 1: Stream statistics

blocks (roughly corresponding with days). We also used the AFP portion of the Giga Word corpus as another source of data that evolves at a slower pace. This data was divided into 50 equal sized blocks. Table 1 gives statistics about the data. As can be seen, the Twitter data is vastly larger than newswire and arrives at a much faster rate.

We considered the following models. Each one (apart from the exact model) was trained using the same amount of data:

- **Static.** This model was trained using data from the start of the duration and never varied. It is a baseline.
- **Exact.** This model was trained using *all* available data from the start of the stream and acts as an upper bound on performance.
- **Moving Window.** This model used all data in a fixed-sized window immediately before the given test point.
- **Uniform.** Here, we use uniform reservoir sampling to select the data.
- **Exponential.** Lastly, we use exponential reservoir sampling to select the data. This model is parameterised, indicating how strongly biased towards the present the sample will be. The β parameter is a multiplier over the reservoir length. For example, a β value of 1.1 with a sample size of 10 means the value is 11. In general, β always needs to be bigger than the reservoir size.

We sample over whole sentences (or Tweets) and not ngrams.⁴ Using ngrams instead would give us a finer-grained control over results, but would come at the expense of greatly complicating the analysis. This is because we would need to reason about not just a set of items but a multiset of items. Note that because the samples are large⁵, variations across samples will be small.

⁴A consequence is that we do not guarantee that each sample uses exactly the same number of grams. This can be tackled by randomly removing sampled sentences.

⁵Each day consists of approximately four million Tweets and we evaluate on a whole day.

Day	Uniform	β value			
	∞	1.1	1.3	1.5	2.0
5	619.4	619.4	619.4	619.4	619.4
6	601.0	601.0	603.8	606.6	611.1
7	603.0	599.4	602.7	605.6	612.1
8	614.6	607.7	611.9	614.3	621.6
9	623.3	611.5	615.0	620.0	628.1
10	656.2	643.1	647.2	650.1	658.0
12	646.6	628.9	633.0	636.5	644.6
15	647.7	628.7	630.4	634.5	641.6
20	636.7	605.3	608.4	610.8	618.4
25	631.5	601.9	603.3	604.4	610.0

Table 2: Perplexities for different β values over Twitter (sample size = five days). Lower is better.

We test the model on unseen data from all of the next day (or block). Afterwards, we advance to the next day (block) and repeat, potentially incorporating the previously seen test data into the current training data. Evaluation is in terms of perplexity (which is standard for language modelling).

We used *KenLM* for building models and evaluating them (Heafield, 2011). Each model was an unpruned trigram, with Kneser-Ney smoothing. Increasing the language model order would not change the results. Here the focus is upon which data is used in a model (that is, which data is added and which data is removed) and not upon making it compact or making retraining efficient.

4.2 Varying the β Parameter

Table 2 shows the effect of varying the β parameter (using Twitter). The higher the β value, the more uniform the sampling. As can be seen, performance improves when sampling becomes more biased. Not shown here, but for Twitter, even smaller β values produce better results and for newswire, results degrade. These differences are small and do not affect any conclusions made here. In practise, this value would be set using a development set and to simplify the rest of the paper, all other experiments use the same β value (1.1).

4.3 Varying the Amount of Data

Does the amount of data used in a model affect results? Table 3 shows the results for Twitter when varying the amount of data in the sample and using exponential sampling ($\beta = 1.1$). In parentheses for each result, we show the corresponding moving window results. As expected, using more data improves results. We see that for each sample size, exponential sampling outperforms our moving window. In the limit, all sampling methods would produce the same results.

Day	Sample Size (Days)		
	1	2	3
5	652.5 (661.2)	629.1 (635.8)	624.8 (625.9)
6	635.4 (651.6)	611.6 (620.8)	604.0 (608.7)
7	636.0 (647.3)	611.0 (625.2)	603.7 (612.5)
8	654.8 (672.7)	625.6 (641.6)	614.6 (626.9)
9	653.9 (662.8)	628.3 (643.0)	618.8 (632.2)
10	679.1 (687.8)	654.3 (666.8)	646.6 (659.7)
12	671.1 (681.9)	645.8 (658.6)	633.8 (647.5)
15	677.7 (697.9)	647.4 (668.0)	636.4 (652.6)
20	648.1 (664.6)	621.4 (637.9)	612.2 (627.6)
25	657.5 (687.5)	625.3 (664.4)	613.4 (641.8)

Table 3: Perplexities for different sample sizes over Twitter. Lower is better.

4.4 Alternative Sampling Strategies

Table 4 compares the two baselines against the two forms of reservoir sampling. For Twitter, we see a clear recency effect. The static baseline gets worse and worse as it recedes from the current test point. Uniform sampling does better, but it in turn is beaten by the Moving Window Model. However, this in turn is beaten by our exponential reservoir sampling.

Day	Static	Moving	Uniform	Exp	Exact
5	619.4	619.4	619.4	619.4	619.4
6	664.8	599.7	601.8	601.0	597.6
7	684.4	602.8	603.0	599.3	595.6
8	710.1	612.0	614.6	607.7	603.5
9	727.0	617.9	623.3	613.0	608.7
10	775.6	651.2	656.2	642.0	640.5
12	776.7	639.0	646.6	628.7	627.5
15	777.1	638.3	647.7	626.7	627.3
20	800.9	619.1	636.7	604.9	607.3
25	801.4	621.7	631.5	601.5	597.6

Table 4: Perplexities for differently selected samples over Twitter (sample size = five days, $\beta = 1.1$). Results in **bold** are the best sampling results. Lower is better.

4.5 GigaWord

Twitter is a fast moving, rapidly changing multilingual stream and it is not surprising that our exponential reservoir sampling proves beneficial. Is it still useful for a more conventional stream that is drawn from a much smaller population of reporters? We repeated our experiments, using the same rolling training and testing evaluation as before, but this time using newswire for data.

Table 5 shows the perplexities when using the Gigaword stream. We see the same general trends, albeit with less of a difference between exponential sampling and our moving window. Perplexity values are all lower than for Twitter.

Block	Static	Moving	Uniform	Exp
11	416.5	381.1	382.0	382.0
15	436.7	353.3	357.5	352.8
20	461.8	347.0	354.4	344.6
25	315.6	214.9	222.2	211.3
30	319.1	200.5	213.5	199.5
40	462.5	304.4	313.2	292.9

Table 5: Perplexities for differently selected samples over Gigaword (sample size = 10 blocks, $\beta = 1.1$). Lower is better.

4.6 Why does this work for Twitter?

Although the perplexity results demonstrate that exponential sampling is on average beneficial, it is useful to analyse the results in more detail. For a large stream size (25 days), we built models using uniform, exponential ($\beta = 1.1$) and our moving window sampling methods. Each approach used the same amount of data. For the same test set (four million Tweets), we computed per-Tweet log likelihoods and looked at the difference between the model that best explained each tweet and the second best model (ie the margin). This gives us an indication of how much a given model better explains a given Tweet. Analysing the results, we found that most gains came from short grams and very few came from entire Tweets being reposted (or retweeted). This suggests that the Twitter results follow previously reported observations on how language can be bursty and not from Twitter-specific properties.

5 Conclusion

We have introduced exponential reservoir sampling as an elegant way to model a stream of unbounded size, yet using fixed space. It naturally allows one to take account of recency effects present in many natural streams. We expect that our language model could improve other Social Media tasks, for example lexical normalisation (Han and Baldwin, 2011) or even event detection (Lin et al., 2011). The approach is fully general and not just limited to language modelling. Future work should look at other distributions for sampling and consider tasks such as machine translation over Social Media.

Acknowledgments This work was carried out when MO was on sabbatical at the HLTCOE and CLSP.

References

- Charu C Aggarwal. 2006. On biased reservoir sampling in the presence of stream evolution. In *Proceedings of the 32nd international conference on Very large data bases*, pages 607–618. VLDB Endowment.
- Doug Beeferman, Adam Berger, and John Lafferty. 1997. A model of lexical attractions and repulsion. 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*, pages 373–380. Association for Computational Linguistics.
- K. Church and W. A. Gale. 1995. Poisson mixtures. *Natural Language Engineering*, 1:163–190.
- Kenneth W Church. 2000. Empirical estimates of adaptation: the chance of two noriegas is closer to $p/2$ than p . In *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pages 180–186. Association for Computational Linguistics.
- Amit Goyal, Hal Daumé III, and Suresh Venkatasubramanian. 2009. Streaming for large scale NLP: Language Modeling. In *Proceedings of NAACL*.
- Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT '11*, pages 368–378, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, United Kingdom, July.
- Raymond Lau, Ronald Rosenfeld, and SaIm Roukos. 1993. Trigger-based language models: A maximum entropy approach. In *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, volume 2, pages 45–48. IEEE.
- Abby Levenberg and Miles Osborne. 2009. Stream-based randomised language models for smt. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 756–764. Association for Computational Linguistics.
- Jimmy Lin, Rion Snow, and William Morgan. 2011. Smoothing techniques for adaptive online language models: topic tracking in tweet streams. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 422–429. ACM.
- David Talbot and Miles Osborne. 2007. Randomised language modelling for statistical machine translation. In *Proceedings of ACL*.
- Benjamin Van Durme and Ashwin Lall. 2009. Probabilistic Counting with Randomized Storage. In *Proceedings of IJCAI*.
- Benjamin Van Durme and Ashwin Lall. 2011. Efficient online locality sensitive hashing via reservoir counting. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 18–23. Association for Computational Linguistics.
- Benjamin Van Durme. 2012. Streaming analysis of discourse participants. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 48–58. Association for Computational Linguistics.
- Jeffrey S. Vitter. 1985. Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11:37–57, March.

A Piece of My Mind: A Sentiment Analysis Approach for Online Dispute Detection

Lu Wang

Department of Computer Science
Cornell University
Ithaca, NY 14853
luwang@cs.cornell.edu

Claire Cardie

Department of Computer Science
Cornell University
Ithaca, NY 14853
cardie@cs.cornell.edu

Abstract

We investigate the novel task of *online dispute detection* and propose a sentiment analysis solution to the problem: we aim to identify the sequence of sentence-level sentiments expressed during a discussion and to use them as features in a classifier that predicts the DISPUTE/NON-DISPUTE label for the discussion as a whole. We evaluate dispute detection approaches on a newly created corpus of Wikipedia Talk page disputes and find that classifiers that rely on our sentiment tagging features outperform those that do not. The best model achieves a very promising F1 score of 0.78 and an accuracy of 0.80.

1 Introduction

As the web has grown in popularity and scope, so has the promise of collaborative information environments for the joint creation and exchange of knowledge (Jones and Rafaeli, 2000; Sack, 2005). Wikipedia, a wiki-based online encyclopedia, is arguably the best example: its distributed editing environment allows readers to collaborate as content editors and has facilitated the production of over four billion articles¹ of surprisingly high quality (Giles, 2005) in English alone since its debut in 2001.

Existing studies of collaborative knowledge systems have shown, however, that the quality of the generated content (e.g. an encyclopedia article) is highly correlated with the effectiveness of the online collaboration (Kittur and Kraut, 2008; Kraut and Resnick, 2012); fruitful collaboration, in turn, inevitably requires dealing with the disputes and conflicts that arise (Kittur et al., 2007). Unfortunately, human monitoring of the often massive social media and collaboration sites to detect, much less mediate, disputes is not feasible.

¹<http://en.wikipedia.org>

In this work, we investigate the heretofore novel task of *dispute detection in online discussions*. Previous work in this general area has analyzed dispute-laden content to discover features correlated with conflicts and disputes (Kittur et al., 2007). Research focused primarily on cues derived from the edit history of the jointly created content (e.g. the number of revisions, their temporal density (Kittur et al., 2007; Yasseri et al., 2012)) and relied on small numbers of manually selected discussions known to involve disputes. In contrast, we investigate methods for the automatic detection, i.e. prediction, of discussions involving disputes. We are also interested in understanding whether, and which, linguistic features of the discussion are important for dispute detection.

Drawing inspiration from studies of human mediation of online conflicts (e.g. Billings and Watts (2010), Kittur et al. (2007), Kraut and Resnick (2012)), we hypothesize that effective methods for dispute detection should take into account the sentiment and opinions expressed by participants in the collaborative endeavor. As a result, we propose a sentiment analysis approach for online dispute detection that identifies the sequence of sentence-level sentiments (i.e. very negative, negative, neutral, positive, very positive) expressed during the discussion and uses them as features in a classifier that predicts the DISPUTE/NON-DISPUTE label for the discussion as a whole. Consider, for example, the snippet in Figure 1 from the Wikipedia Talk page for the article on Philadelphia; it discusses the choice of a picture for the article’s “infobox”. The sequence of almost exclusively negative statements provides evidence of a dispute in this portion of the discussion.

Unfortunately, sentence-level sentiment tagging for this domain is challenging in its own right due to the less formal, often ungrammatical, language and the dynamic nature of online conversations. “*Really, grow up*” (segment 3) should

1-**Emy111**: I think everyone is forgetting that my previous image was the lead image for well over a year! ...
 > **Massimo**: I'm sorry to say so, but it is grossly over processed...
 2-**Emy111**: i'm glad you paid more money for a camera than I did. **congrats...** i appreciate your constructive criticism. **thank you**.
 > **Massimo**: I just want to have the best picture as a lead for the article ...
 3-**Emy111**: Wow, I am really enjoying this photography debate... [so don't make assumptions you know nothing about.]_{NN} [Really, grow up.]_N [If you all want to complain about Photoshop editing, lets all go buy medium format film cameras, shoot film, and scan it, so no manipulation is possible.]_o [Sound good?]_{NN}
 > **Massimo**: ... I do feel it is a pity, that you turned out to be a sore loser...

Figure 1: From the Wikipedia Talk page for the article “Philadelphia”. Omitted sentences are indicated by ellipsis. Names of editors are in **bold**. The start of each set of related turns is numbered; “>” is an indicator for the reply structure.

presumably be tagged as a negative sentence as should the sarcastic sentences “*Sounds good?*” (in the same turn) and “*congrats*” and “*thank you*” (in segment 2). We expect that these, and other, examples will be difficult for the sentence-level classifier unless the discourse context of each sentence is considered. Previous research on sentiment prediction for online discussions, however, focuses on turn-level predictions (Hahn et al., 2006; Yin et al., 2012).² As the first work that predicts sentence-level sentiment for online discussions, we investigate isotonic Conditional Random Fields (CRFs) (Mao and Lebanon, 2007) for the sentiment-tagging task as they preserve the advantages of the popular CRF-based sequential tagging models (Lafferty et al., 2001) while providing an efficient mechanism for encoding domain knowledge — in our case, a sentiment lexicon — through isotonic constraints on model parameters.

We evaluate our dispute detection approach using a newly created corpus of discussions from Wikipedia Talk pages (3609 disputes, 3609 non-disputes).³ We find that classifiers that employ the learned sentiment features outperform others that do not. The best model achieves a very promising F1 score of 0.78 and an accuracy of 0.80 on the Wikipedia dispute corpus. To the best of our knowledge, this represents the first computational approach to automatically identify online disputes on a dataset of scale.

Additional Related Work. Sentiment analysis has been utilized as a key enabling technique in a number of conversation-based applications. Previous work mainly studies the attitudes in spoken

²A notable exception is Hassan et al. (2010), which identifies sentences containing “attitudes” (e.g. opinions), but does not distinguish them w.r.t. sentiment. Context information is also not considered.

³The talk page associated with each article records conversations among editors about the article content and allows editors to discuss the writing process, e.g. planning and organizing the content.

meetings (Galley et al., 2004; Hahn et al., 2006) or broadcast conversations (Wang et al., 2011) using variants of Conditional Random Fields (Lafferty et al., 2001) and predicts sentiment at the turn-level, while our predictions are made for each sentence.

2 Data Construction: A Dispute Corpus

We construct the first dispute detection corpus to date; it consists of dispute and non-dispute discussions from Wikipedia Talk pages.

Step 1: Get Talk Pages of Disputed Articles.

Wikipedia articles are edited by different editors. If an article is observed to have disputes on its *talk page*, editors can assign dispute tags to the article to flag it for attention. In this research, we are interested in talk pages whose corresponding articles are labeled with the following tags: DISPUTED, TOTALLYDISPUTED, DISPUTED-SECTION, TOTALLYDISPUTED-SECTION, POV. The tags indicate that an article is disputed, or the neutrality of the article is disputed (POV).

We use the 2013-03-04 Wikipedia data dump, and extract talk pages for articles that are labeled with dispute tags by checking the revision history. This results in 19,071 talk pages.

Step 2: Get Discussions with Disputes.

Dispute tags can also be added to *talk pages* themselves. Therefore, in addition to the tags mentioned above, we also consider the “Request for Comment” (RFC) tag on talk pages. According to Wikipedia⁴, RFC is used to request outside opinions concerning the disputes.

3609 discussions are collected with dispute tags found in the revision history. We further classify dispute discussions into three subcategories: CONTROVERSY, REQUEST FOR COMMENT (RFC), and RESOLVED based on the tags found in discussions (see Table 1). The numbers of discussions for the three types are 42, 3484, and 105, respectively. Note that dispute tags only appear in a small number of articles and talk pages. There may exist other discussions with disputes.

Dispute Subcategory	Wikipedia Tags on Talk pages
Controversy	CONTROVERSIAL, TOTALLYDISPUTED, DISPUTED, CALM TALK, POV
Request for Comment	RFC
Resolved	Any tag from above + RESOLVED

Table 1: Subcategory for disputes with corresponding tags. Note that each discussion in the RESOLVED class has more than one tag.

Step 3: Get Discussions without Disputes. Likewise, we collect non-dispute discussions from

⁴http://en.wikipedia.org/wiki/Wikipedia:Requests_for_comment

pages that are never tagged with disputes. We consider non-dispute discussions with at least 3 distinct speakers and 10 turns. 3609 discussions are randomly selected with this criterion. The average turn numbers for dispute and non-dispute discussions are 45.03 and 22.95, respectively.

3 Sentence-level Sentiment Prediction

This section describes our sentence-level sentiment tagger, from which we construct features for dispute detection (Section 4).

Consider a discussion comprised of sequential turns; each turn consists of a sequence of sentences. Our model takes as input the sentences $\mathbf{x} = \{x_1, \dots, x_n\}$ from a single turn, and outputs the corresponding sequence of sentiment labels $\mathbf{y} = \{y_1, \dots, y_n\}$, where $y_i \in \mathcal{O}$, $\mathcal{O} = \{\text{NN}, \text{N}, \text{O}, \text{P}, \text{PP}\}$. The labels in \mathcal{O} represent very negative (NN), negative (N), neutral (O), positive (P), and very positive (PP), respectively.

Given that traditional Conditional Random Fields (CRFs) (Lafferty et al., 2001) ignore the ordinal relations among sentiment labels, we choose *isotonic CRFs* (Mao and Lebanon, 2007) for sentence-level sentiment analysis as they can enforce monotonicity constraints on the parameters consistent with the ordinal structure and domain knowledge (e.g. word-level sentiment conveyed via a lexicon). Concretely, we take a lexicon $\mathcal{M} = \mathcal{M}_p \cup \mathcal{M}_n$, where \mathcal{M}_p and \mathcal{M}_n are two sets of features (usually words) identified as strongly associated with positive and negative sentiment. Assume $\mu_{\langle\sigma,w\rangle}$ encodes the weight between label σ and feature w , for each feature $w \in \mathcal{M}_p$; then the isotonic CRF enforces $\sigma \leq \sigma' \Rightarrow \mu_{\langle\sigma,w\rangle} \leq \mu_{\langle\sigma',w\rangle}$. For example, when “totally agree” is observed in training, parameter $\mu_{\langle\text{PP}, \text{totally agree}\rangle}$ is likely to increase. Similar constraints are defined on \mathcal{M}_n .

Our lexicon is built by combining MPQA (Wilson et al., 2005), General Inquirer (Stone et al., 1966), and SentiWordNet (Esuli and Sebastiani, 2006) lexicons. Words with contradictory sentiments are removed. We use the features in Table 2 for sentiment prediction.

Syntactic/Semantic Features. We have two versions of dependency relation features, the original form and a form that generalizes a word to its POS tag, e.g. “nsubj(wrong, you)” is generalized to “nsubj(ADJ, you)” and “nsubj(wrong, PRP)”.

Discourse Features. We extract the initial unigram, bigram, and trigram of each utterance as dis-

Lexical Features - unigram/bigram - number of words all uppercased - number of words Discourse Features - initial uni-/bi-/tri-gram - repeated punctuations - hedging phrases collected from Farkas et al. (2010) - number of negators	Syntactic/Semantic Features - unigram with POS tag - dependency relation Conversation Features - quote overlap with target - TFIDF similarity with target (remove quote first) Sentiment Features - connective + sentiment words - sentiment dependency relation - sentiment words
--	--

Table 2: Features used in sentence-level sentiment prediction. Numerical features are first normalized by standardization, then binned into 5 categories.

course features (Hirschberg and Litman, 1993).

Sentiment Features. We gather connectives from the Penn Discourse TreeBank (Rashmi Prasad and Webber, 2008) and combine them with any sentiment word that precedes or follows it as new features. Sentiment dependency relations are the dependency relations that include a sentiment word. We replace those words with their polarity equivalents. For example, relation “nsubj(wrong, you)” becomes “nsubj(SentiWord_{neg}, you)”.

4 Online Dispute Detection

4.1 Training A Sentiment Classifier

Dataset. We train the sentiment classifier using the *Authority and Alignment in Wikipedia Discussions (AAWD)* corpus (Bender et al., 2011) on a 5-point scale (i.e. NN, N, O, P, PP). AAWD consists of 221 English Wikipedia discussions with positive and negative alignment annotations. Annotators either label each sentence as positive, negative or neutral, or label the full turn. For instances that have only a turn-level label, we assume all sentences have the same label as the turn. We further transform the labels into the five sentiment labels. Sentences annotated as being a positive alignment by at least two annotators are treated as very positive (PP). If a sentence is only selected as positive by one annotator or obtains the label via turn-level annotation, it is positive (P). Very negative (NN) and negative (N) are collected in the same way. All others are neutral (O). Among all 16,501 sentences in AAWD, 1,930 and 1,102 are labeled as NN and N. 532 and 99 of them are PP and P. The other 12,648 are considered neutral.

Evaluation. To evaluate the performance of the sentiment tagger, we compare to two baselines. (1) **Baseline (Polarity):** a sentence is predicted as positive if it has more positive words than negative words, or negative if more negative words are observed. Otherwise, it is neutral. (2) **Baseline (Distance)** is extended from (Hassan et al., 2010). Each sentiment word is associated with the closest

	Pos	Neg	Neutral
Baseline (Polarity)	22.53	38.61	66.45
Baseline (Distance)	33.75	55.79	88.97
SVM (3-way)	44.62	52.56	80.84
CRF (3-way)	56.28	56.37	89.41
CRF (5-way)	58.39	56.30	90.10
isotonic CRF	68.18	62.53	88.87

Table 3: F1 scores for positive and negative alignment on Wikipedia Talk pages (AAWD) using 5-fold cross-validation. In each column, **bold** entries (if any) are statistically significantly higher than all the rest. We also compare with an SVM and linear CRF trained with three classes (3-way). Our model based on the isotonic CRF produces significantly better results than all the other systems.

second person pronoun, and a surface distance is computed. An SVM classifier (Joachims, 1999) is trained using features of the sentiment words and minimum/maximum/average of the distances.

We also compare with two state-of-the-art methods that are used in sentiment prediction for conversations: (1) an SVM (RBF kernel) that is employed for identifying sentiment-bearing sentences (Hassan et al., 2010), and (dis)agreement detection (Yin et al., 2012) in online debates; (2) a Linear CRF for (dis)agreement identification in broadcast conversations (Wang et al., 2011).

We evaluate the systems using standard F1 on classes of positive, negative, and neutral, where samples predicted as PP and P are positive alignment, and samples tagged as NN and N are negative alignment. Table 3 describes the main results on the AAWD dataset: our isotonic CRF based system significantly outperforms the alternatives for positive and negative alignment detection (paired- t test, $p < 0.05$).

4.2 Dispute Detection

We model dispute detection as a standard binary classification task, and investigate four major types of features as described below.

Lexical Features. We first collect unigram and bigram features for each discussion.

Topic Features. Articles on specific topics, such as politics or religions, tend to arouse more disputes. We thus extract the category information of the corresponding article for each talk page. We further utilize unigrams and bigrams of the category as topic features.

Discussion Features. This type of feature aims to capture the structure of the discussion. Intuitively, the more turns or the more participants a discussion has, the more likely there is a dispute. Meanwhile, participants tend to produce longer utterances when they make arguments. We choose number of turns, number

of participants, average number of words in each turn as features. In addition, the frequency of revisions made during the discussion has been shown to be good indicator for controversial articles (Vuong et al., 2008), that are presumably prone to have disputes. Therefore, we encode the number of revisions that happened during the discussion as a feature.

Sentiment Features. This set of features encode the sentiment distribution and transition in the discussion. We train our sentiment tagging model on the full AAWD dataset, and run it on the Wikipedia dispute corpus.

Given that consistent negative sentiment flow usually indicates an ongoing dispute, we first extract features from sentiment distribution in the form of number/probability of sentiment per type. We also estimate the sentiment transition probability $P(S_t \rightarrow S_{t+1})$ from our predictions, where S_t and S_{t+1} are sentiment labels for the current sentence and the next. We then have features as number/portion of sentiment transitions per type.

Features described above mostly depict the *global* sentiment flow in the discussions. We further construct a *local* version of them, since sentiment distribution may change as discussion proceeds. For example, less positive sentiment can be observed as dispute being escalated. We thus split each discussion into three equal length stages, and create sentiment distribution and transition features for each stage.

	Prec	Rec	F1	Acc
Baseline (Random)	50.00	50.00	50.00	50.00
Baseline (All dispute)	50.00	100.00	66.67	50.00
Logistic Regression	74.76	72.29	73.50	73.94
SVM _{Linear}	69.81	71.90	70.84	70.41
SVM _{RBF}	77.38	79.14	78.25	80.00

Table 4: Dispute detection results on Wikipedia Talk pages. The numbers are multiplied by 100. The items in **bold** are statistically significantly higher than others in the same column (paired- t test, $p < 0.05$). SVM with the RBF kernel achieves the best performance in precision, F1, and accuracy.

Results and Error Analysis. We experiment with logistic regression, SVM with linear and RBF kernels, which are effective methods in multiple text categorization tasks (Joachims, 1999; Zhang and J. Oles, 2001). We normalize the features by standardization and conduct a 5-fold cross-validation. Two baselines are listed: (1) labels are randomly assigned; (2) all discussions have disputes.

Main results for different classifiers are displayed in Table 4. All learning based methods

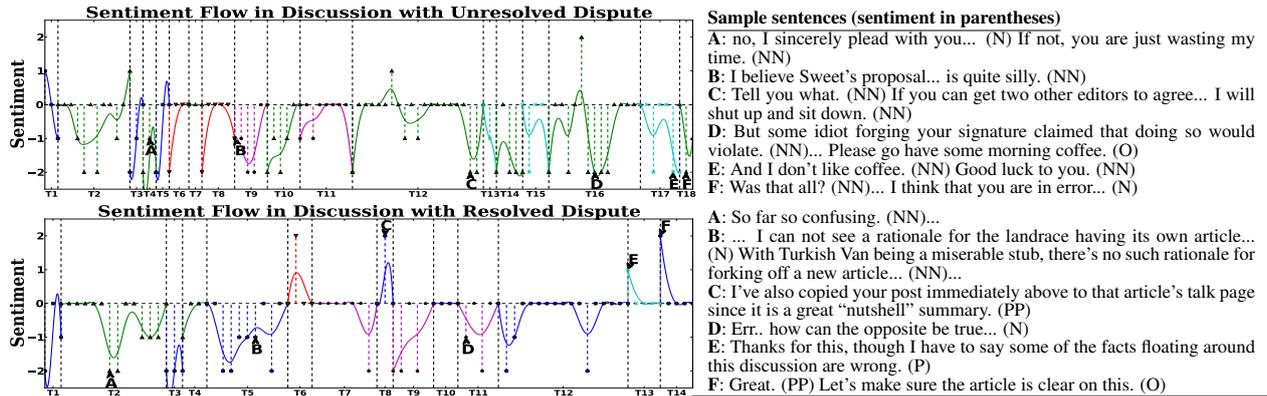


Figure 2: Sentiment flow for a discussion with **unresolved** dispute about the definition of “white people” (top) and a discussion with **resolved** dispute on merging articles about van cat (bottom). The labels {NN, N, O, P, PP} are mapped to $\{-2, -1, 0, 1, 2\}$ in sequence. Sentiment values are convolved by Gaussian smoothing kernel, and cubic-spline interpolation is then conducted. Different speakers are represented by curves of different colors. Dashed vertical lines delimit turns. Representative sentences are labeled with letters and their sentiment labels are shown on the right. For unresolved dispute (top), we see that negative sentiment exists throughout the discussion. Whereas, for the resolved dispute (bottom), less negative sentiment is observed at the end of the discussion; participants also show appreciation after the problem is solved (e.g. E and F in the plot).

	Prec	Rec	F1	Acc
Lexical (Lex)	75.86	34.66	47.58	61.82
Topic (Top)	68.44	71.46	69.92	69.26
Discussion (Dis)	69.73	76.14	72.79	71.54
Sentiment (Senti _{g+l})	72.54	69.52	71.00	71.60
Top + Dis	68.49	71.79	70.10	69.38
Top + Dis + Senti _g	77.39	78.36	77.87	77.74
Top + Dis + Senti _{g+l}	77.38	79.14	78.25	80.00
Lex + Top + Dis + Senti _{g+l}	78.38	75.12	76.71	77.20

Table 5: Dispute detection results with different feature sets by SVM with RBF kernel. The numbers are multiplied by 100. Senti_g represents global sentiment features, and Senti_{g+l} includes both global and local features. The number in **bold** is statistically significantly higher than other numbers in the same column (paired-*t* test, $p < 0.05$), and the *italic* entry has the highest absolute value.

outperform the two baselines, and among them, SVM with the RBF kernel achieves the best F1 score and accuracy (0.78 and 0.80). Experimental results with various combinations of features sets are displayed in Table 5. As it can be seen, sentiment features obtains the best accuracy among the four types of features. A combination of topic, discussion, and sentiment features achieves the best performance on recall, F1, and accuracy. Specifically, the accuracy is significantly higher than all the other systems (paired-*t* test, $p < 0.05$).

After a closer look at the results, we find two main reasons for incorrect predictions. Firstly, sentiment prediction errors get propagated into dispute detection. Due to the limitation of existing general-purpose lexicons, some opinionated dialog-specific terms are hard to catch. For example, “I told you over and over again...” strongly suggests a negative sentiment, but no single word shows negative connotation. Constructing a lexicon tuned for conversational text may improve the performance. Secondly, some dispute discussions are harder to detect than the others due to differ-

ent dialog structures. For instance, the recalls for dispute discussions of “controversy”, “RFC”, and “resolved” are 0.78, 0.79, and 0.86 respectively. We intend to design models that are able to capture dialog structures in the future work.

Sentiment Flow Visualization. We visualize the sentiment flow of two disputed discussions in Figure 2. The plots reveal persistent negative sentiment in unresolved disputes (top). For the resolved dispute (bottom), participants show gratitude when the problem is settled.

5 Conclusion

We present a sentiment analysis-based approach to online dispute detection. We create a large-scale dispute corpus from Wikipedia Talk pages to study the problem. A sentiment prediction model based on isotonic CRFs is proposed to output sentiment labels at the sentence-level. Experiments on our dispute corpus also demonstrate that classifiers trained with sentiment tagging features outperform others that do not.

Acknowledgments We heartily thank the Cornell NLP Group, the reviewers, and Yiye Ruan for helpful comments. We also thank Emily Bender and Mari Ostendorf for providing the AAWD dataset. This work was supported in part by NSF grants IIS-0968450 and IIS-1314778, and DARPA DEFT Grant FA8750-13-2-0015. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF, DARPA or the U.S. Government.

References

- Emily M. Bender, Jonathan T. Morgan, Meghan Oxley, Mark Zachry, Brian Hutchinson, Alex Marin, Bin Zhang, and Mari Ostendorf. 2011. Annotating social acts: Authority claims and alignment moves in wikipedia talk pages. In *Proceedings of the Workshop on Languages in Social Media, LSM '11*, pages 48–57, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Matt Billings and Leon Adam Watts. 2010. Understanding dispute resolution online: using text to reflect personal and substantive issues in conflict. In Elizabeth D. Mynatt, Don Schoner, Geraldine Fitzpatrick, Scott E. Hudson, W. Keith Edwards, and Tom Rodden, editors, *CHI*, pages 1447–1456. ACM.
- Andrea Esuli and Fabrizio Sebastiani. 2006. Sentimentnet: A publicly available lexical resource for opinion mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC06)*, pages 417–422.
- Richárd Farkas, Veronika Vincze, György Móra, János Csirik, and György Szarvas. 2010. The conll-2010 shared task: Learning to detect hedges and their scope in natural language text. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning — Shared Task, CoNLL '10: Shared Task*, pages 1–12, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Michel Galley, Kathleen McKeown, Julia Hirschberg, and Elizabeth Shriberg. 2004. Identifying agreement and disagreement in conversational speech: use of Bayesian networks to model pragmatic dependencies. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 669+, Morristown, NJ, USA. Association for Computational Linguistics.
- Jim Giles. 2005. Internet encyclopaedias go head to head. *Nature*, 438(7070):900–901.
- Sangyun Hahn, Richard Ladner, and Mari Ostendorf. 2006. Agreement/disagreement classification: Exploiting unlabeled data using contrast classifiers. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 53–56, New York City, USA, June. Association for Computational Linguistics.
- Ahmed Hassan, Vahed Qazvinian, and Dragomir Radev. 2010. What’s with the attitude?: Identifying sentences with attitude in online discussions. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10*, pages 1245–1255, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Julia Hirschberg and Diane Litman. 1993. Empirical studies on the disambiguation of cue phrases. *Comput. Linguist.*, 19(3):501–530, September.
- Thorsten Joachims. 1999. Advances in kernel methods. chapter Making Large-scale Support Vector Machine Learning Practical, pages 169–184. MIT Press, Cambridge, MA, USA.
- Quentin Jones and Sheizaf Rafaeli. 2000. Time to split, virtually: discourse architecture and community building create vibrant virtual publics. *Electronic Markets*, 10:214–223.
- Aniket Kittur and Robert E. Kraut. 2008. Harnessing the wisdom of crowds in wikipedia: Quality through coordination. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work, CSCW '08*, pages 37–46, New York, NY, USA. ACM.
- Aniket Kittur, Bongwon Suh, Bryan A. Pendleton, and Ed H. Chi. 2007. He says, she says: Conflict and coordination in wikipedia. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '07*, pages 453–462, New York, NY, USA. ACM.
- Robert E. Kraut and Paul Resnick. 2012. *Building successful online communities: Evidence-based social design*. MIT Press, Cambridge, MA.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01*, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Yi Mao and Guy Lebanon. 2007. Isotonic conditional random fields and local sentiment flow. In *Advances in Neural Information Processing Systems*.
- Alan Lee Eleni Miltsakaki Livio Robaldo Aravind Joshi Rashmi Prasad, Nikhil Dinesh and Bonnie Webber. 2008. The penn discourse treebank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may. European Language Resources Association (ELRA). <http://www.lrec-conf.org/proceedings/lrec2008/>.
- Warren Sack. 2005. Digital formations: It and new architectures in the global realm. chapter Discourse architecture and very large-scale conversation, pages 242–282. Princeton University Press, Princeton, NJ USA.
- Philip J. Stone, Dexter C. Dunphy, Marshall S. Smith, and Daniel M. Ogilvie. 1966. *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press, Cambridge, MA.
- Ba-Quy Vuong, Ee-Peng Lim, Aixin Sun, Minh-Tam Le, Hady Wirawan Lauw, and Kuiyu Chang. 2008. On ranking controversies in wikipedia: Models and evaluation. In *Proceedings of the 2008 International Conference on Web Search and Data Mining, WSDM '08*, pages 171–182, New York, NY, USA. ACM.

- Wen Wang, Sibel Yaman, Kristin Precoda, Colleen Richey, and Geoffrey Raymond. 2011. Detection of agreement and disagreement in broadcast conversations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 374–378, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 347–354, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Taha Yasseri, Róbert Sumi, András Rung, András Kornai, and János Kertész. 2012. Dynamics of conflicts in wikipedia. *CoRR*, abs/1202.3643.
- Jie Yin, Paul Thomas, Nalin Narang, and Cecile Paris. 2012. Unifying local and global agreement and disagreement classification in online debates. In *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, WASSA '12, pages 61–69, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Tong Zhang and Frank J. Oles. 2001. Text categorization based on regularized linear classification methods. *Inf. Retr.*, 4(1):5–31, April.

A Simple Bayesian Modelling Approach to Event Extraction from Twitter

Deyu Zhou^{†‡} Liangyu Chen[†] Yulan He[§]

[†] School of Computer Science and Engineering, Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University, China

[‡] State Key Laboratory for Novel Software Technology, Nanjing University, China

[§] School of Engineering and Applied Science, Aston University, UK

d.zhou@seu.edu.cn, clylcn@126.com, y.he@cantab.net

Abstract

With the proliferation of social media sites, social streams have proven to contain the most up-to-date information on current events. Therefore, it is crucial to extract events from the social streams such as tweets. However, it is not straightforward to adapt the existing event extraction systems since texts in social media are fragmented and noisy. In this paper we propose a simple and yet effective Bayesian model, called Latent Event Model (LEM), to extract structured representation of events from social media. LEM is fully unsupervised and does not require annotated data for training. We evaluate LEM on a Twitter corpus. Experimental results show that the proposed model achieves 83% in F-measure, and outperforms the state-of-the-art baseline by over 7%.

1 Introduction

Event extraction is to automatically identify events from text with information about *what* happened, *when*, *where*, to *whom*, and *why*. Previous work in event extraction has focused largely on news articles, as the newswire texts have been the best source of information on current events (Hogenboom et al., 2011). Approaches for event extraction include knowledge-based (Piskorski et al., 2007; Tanev et al., 2008), data-driven (Piskorski et al., 2008) and a combination of the above two categories (Grishman et al., 2005). Knowledge-based approaches often rely on linguistic and lexicographic patterns which represent expert domain knowledge for particular event types. They lack the flexibility of porting to new domains since extraction patterns often need to be re-defined. Data-driven approaches require large annotated data to train statistical models that approximate linguistic

phenomena. Nevertheless, it is expensive to obtain annotated data in practice.

With the increasing popularity of social media, social networking sites such as Twitter have become an important source of event information. As reported in (Petrovic et al., 2013), even 1% of the public stream of Twitter contains around 95% of all the events reported in the newswire. Nevertheless, the social stream data such as Twitter data pose new challenges. Social media messages are often short and evolve rapidly over time. As such, it is not possible to know the event types a priori and hence violates the use of existing event extraction approaches.

Approaches to event extraction from Twitter make use of a graphical model to extract canonical entertainment events from tweets by aggregating information across multiple messages (Benson et al., 2011). In (Liu et al., 2012), social events involving two persons are extracted from multiple similar tweets using a factor graph by harvesting the redundancy in tweets. Ritter et al. (2012) presented a system called TwiCal which extracts an open-domain calendar of significant events represented by a 4-tuple set including a named entity, event phrase, calendar date, and event type from Twitter.

In our work here, we notice a very important property in social media data that the same event could be referenced by high volume messages. This property allows us resort to statistical models that can group similar events based on the co-occurrence patterns of their event elements. Here, event elements include named entities such as person, company, organization, date/time, location, and the relations among them. We can treat an event as a latent variable and model the generation of an event as a joint distribution of its individual event elements. We thus propose a Latent Event Model (LEM) which can automatically detect events from social media without the use of labeled data.

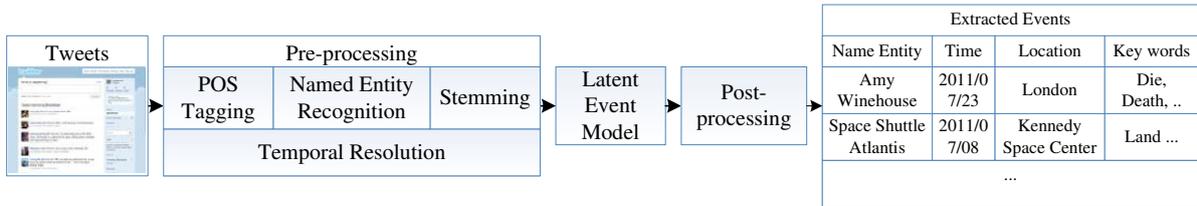


Figure 1: The proposed framework for event extraction from tweets.

Our work is similar to TwiCal in the sense that we also focus on the extraction of structured representation of events from Twitter. However, TwiCal relies on a supervised sequence labeler trained on tweets annotated with event mentions for the identification of event-related phrases. We propose a simple Bayesian modelling approach which is able to directly extract event-related keywords from tweets without supervised learning. Also, TwiCal uses G^2 test to choose an entity y with the strongest association with a date d to form a binary tuple $\langle y, d \rangle$ to represent an event. On the contrary, the structured representation of events can be directly extracted from the output of our LEM model. We have conducted experiments on a Twitter corpus and the results show that our proposed approach outperforms TwiCal, the state-of-the-art open event extraction system, by 7.7% in F-measure.

2 Methodology

Events extracted in our proposed framework are represented as a 4-tuple $\langle y, d, l, k \rangle$, where y stands for a non-location named entity, d for a date, l for a location, and k for an event-related keyword. Each event mentioned in tweets can be closely depicted by this representation. It should be noted that for some events, one or more elements in their corresponding tuples might be absent as their related information is not available in tweets. As illustrated in Figure 1, our proposed framework consists of three main steps, pre-processing, event extraction based on the LEM model and post-processing. The details of our proposed framework are described below.

2.1 Pre-processing

Tweets are pre-processed by time expression recognition, named entity recognition, POS tagging and stemming.

Time Expression Recognition. Twitter users might represent the same date in various forms.

For example, “tomorrow”, “next Monday”, “August 23th” in tweets might all refer to the same day, depending on the date that users wrote the tweets. To resolve the ambiguity of the time expressions, SUTime¹ (Chang and Manning, 2012) is employed, which takes text and a reference date as input and outputs a more accurate date which the time expression refers to.

Named Entity Recognition. Named entity recognition (NER) is a crucial step since the results would directly impact the final extracted 4-tuple $\langle y, d, l, k \rangle$. It is not easy to accurately identify named entities in the Twitter data since tweets contain a lot of misspellings and abbreviations. However, it is often observed that events mentioned in tweets are also reported in news articles in the same period (Petrovic et al., 2013). Therefore, named entities mentioned in tweets are likely to appear in news articles as well. We thus perform named entity recognition in the following way. First, a traditional NER tool such as the Stanford Named Entity Recognizer² is used to identify named entities from the news articles crawled from BBC and CNN during the same period that the tweets were published. The recognised named entities from news are then used to build a dictionary. Named entities from tweets are extracted by looking up the dictionary through fuzzy matching. We have also used a named entity tagger trained specifically on the Twitter data³ (Ritter et al., 2011) to directly extract named entities from tweets. However, as will be shown in Section 3 that using our constructed dictionary for named entity extraction gives better results. We distinguish between location entities, denoted as l , and non-location entities such as person or organization, denoted as y .

¹<http://nlp.stanford.edu/software/sutime.shtml>

²<http://nlp.stanford.edu/software/CRF-NER.shtml>

³<http://github.com/aritter/twitter-nlp>

Finally, we use a POS tagger⁴ trained on tweets (Gimpel et al., 2011) to perform POS tagging on the tweets data and apart from the previously recognised named entities, only words tagged with nouns, verbs or adjectives are kept. These remaining words are subsequently stemmed and words occurred less than 3 times are filtered.

After the pre-processing step, non-location entities y , locations l , dates d and candidate keywords of the tweets are collected as the input to the LEM model for event extraction.

2.2 Event Extraction using the Latent Event Model (LEM)

We propose an unsupervised latent variable model, called the Latent Event Model (LEM), to extract events from tweets. The graphical model of LEM is shown in Figure 2.

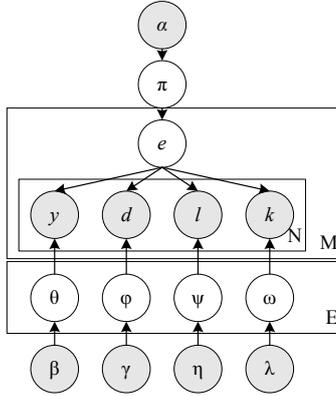


Figure 2: Laten Event Model (LEM).

In this model, we assume that each tweet message $m \in \{1..M\}$ is assigned to one event instance e , while e is modeled as a joint distribution over the named entities y , the date/time d when the event occurred, the location l where the event occurred and the event-related keywords k . This assumption essentially encourages events that involve the same named entities, occur at the same time and in the same location and have similar keyword to be assigned with the same event.

The generative process of LEM is shown below.

- Draw the event distribution $\pi_e \sim \text{Dirichlet}(\alpha)$
- For each event $e \in \{1..E\}$, draw multinomial distributions $\theta_e \sim \text{Dirichlet}(\beta)$, $\varphi_e \sim \text{Dirichlet}(\gamma)$, $\psi_e \sim \text{Dirichlet}(\eta)$, $\omega_e \sim \text{Dirichlet}(\lambda)$.

- For each tweet w

- Choose an event $e \sim \text{Multinomial}(\pi)$,
- For each named entity occur in tweet w , choose a named entity $y \sim \text{Multinomial}(\theta_e)$,
- For each date occur in tweet w , choose a date $d \sim \text{Multinomial}(\varphi_e)$,
- For each location occur in tweet w , choose a location $l \sim \text{Multinomial}(\psi_e)$,
- For other words in tweet w , choose a word $k \sim \text{Multinomial}(\omega_e)$.

We use Collapsed Gibbs Sampling (Griffiths and Steyvers, 2004) to infer the parameters of the model and the latent class assignments for events, given observed data \mathcal{D} and the total likelihood. Gibbs sampling allows us repeatedly sample from a Markov chain whose stationary distribution is the posterior of e_m from the distribution over that variable given the current values of all other variables and the data. Such samples can be used to empirically estimate the target distribution. Letting the subscript $-m$ denote a quantity that excludes data from m th tweet, the conditional posterior for e_m is:

$$P(e_m = t | e_{-m}, \mathbf{y}, \mathbf{d}, \mathbf{l}, \mathbf{z}, \Lambda) \propto \frac{n_t^{-m} + \alpha}{M + E\alpha} \times \prod_{y=1}^Y \frac{\prod_{b=1}^{n_{t,y}^{(m)}} (n_{t,y} - b + \beta)}{\prod_{b=1}^{n_t^{(m)}} (n_t - b + Y\beta)} \times \prod_{d=1}^D \frac{\prod_{b=1}^{n_{t,d}^{(m)}} (n_{t,d} - b + \gamma)}{\prod_{b=1}^{n_t^{(m)}} (n_t - b + D\gamma)} \times \prod_{l=1}^L \frac{\prod_{b=1}^{n_{t,l}^{(m)}} (n_{t,l} - b + \eta)}{\prod_{b=1}^{n_t^{(m)}} (n_t - b + L\eta)} \times \prod_{k=1}^V \frac{\prod_{b=1}^{n_{t,k}^{(m)}} (n_{t,k} - b + \lambda)}{\prod_{b=1}^{n_t^{(m)}} (n_t - b + V\lambda)}$$

where n_t is the number of tweets that have been assigned to the event t ; M is the total number of tweets, $n_{t,y}$ is the number of times named entity y has been associated with event t ; $n_{t,d}$ is the number of times dates d has been associated with event t ; $n_{t,l}$ is the number of times locations l has been assigned with event t ; $n_{t,k}$ is the number of times keyword k has associated with event t , counts with (m) notation denote the counts relating to tweet m only. Y, D, L, V are the total numbers of distinct named entities, dates, locations, and words appeared in the whole Twitter corpus respectively. E is the total number of events which needs to be set.

Once the class assignments for all events are known, we can easily estimate the model parameters $\{\pi, \theta, \varphi, \psi, \omega\}$. We set the hyperparameters $\alpha = \beta = \gamma = \eta = \lambda = 0.5$ and run Gibbs

⁴<http://www.ark.cs.cmu.edu/TweetNLP>

sampler for 10,000 iterations and stop the iteration once the log-likelihood of the training data converges under the learned model. Finally we select an entity, a date, a location, and the top 2 keywords of the highest probability of every event to form a 4-tuple as the representation of that event.

2.3 Post-processing

To improve the precision of event extraction, we remove the least confident event element from the 4-tuples using the following rule. If $P(\text{element})$ is less than $\frac{1}{\xi}P(S)$, where $P(S)$ is the sum of probabilities of the other three elements and ξ is a threshold value and is set to 5 empirically, the element will be removed from the extracted results.

3 Experiments

In this section, we first describe the Twitter corpus used in our experiments and then present how we build a baseline based on the previously proposed TwiCal system (Ritter et al., 2012), the state-of-the-art open event extraction system on tweets. Finally, we present our experimental results.

3.1 Dataset

We use the First Story Detection (FSD) dataset (Petrovic et al., 2013) in our experiment. It consists of 2,499 tweets which are manually annotated with the corresponding event instances resulting in a total of 27 events. The tweets were published between 7th July and 12th September 2011. These events cover a range of categories, from celebrity news to accidents, and from natural disasters to science discoveries. It should be noted here that some event elements such as location is not always available in the tweets. Automatically inferring geolocation of the tweets is a challenging task and will be considered in our future work. For the tweets without time expressions, we used the tweets' publication dates as a default. The number of tweets for each event ranges from 2 to around 1000. We believe that in reality, events which are mentioned in very few tweets are less likely to be significant. Therefore, the dataset was filtered by removing the events which are mentioned in less than 10 tweets. This results in a final dataset containing 2468 tweets annotated with 21 events.

3.2 Baseline construction

The baseline we chose is TwiCal (Ritter et al., 2012). The events extracted in the baseline are

represented as a 3-tuple $\langle y, d, k \rangle^5$, where y stands for a non-location named entity, d for a date and k for an event phrase. We re-implemented the system and evaluate the performance of the baseline on the correctness of the extracted three elements excluding the location element. In the baseline approach, the tuple $\langle y, d, k \rangle$ are extracted in the following ways. Firstly, a named entity recognizer (Ritter et al., 2011) is employed to identify named entities. The TempEx (Mani and Wilson, 2000) is used to resolve temporal expressions. For each date, the baseline approach chose the entity y with the strongest association with the date and form the binary tuple $\langle y, d \rangle$ to represent an event. An event phrase extractor trained on annotated tweets is required to extract event-related phrases. Due to the difficulties of re-implementing the sequence labeler without knowing the actual features set and the annotated training data, we assume all the event-related phrases are identified correctly and simply use the event trigger words annotated in the FSD corpus as k to form the event 3-tuples. It is worth noting that the F-measure reported for the event phrase extraction is only 64% in the baseline approach (Ritter et al., 2012).

3.3 Evaluation Metric

To evaluate the performance of the proposed approach, we use *precision*, *recall*, and *F-measure* as in general information extraction systems (Makhoul et al., 1999). For the 4-tuple $\langle y, d, l, k \rangle$, the precision is calculated based on the following criteria:

1. Do the entity y , location l and date d that we have extracted refer to the same event?
2. Are the keywords k in accord with the event that other extracted elements y, l, d refer to and are they informative enough to tell us what happened?

If the extracted representation does not contain keywords, its precision is calculated by checking the criteria 1. If the extracted representation contains keywords, its precision is calculated by checking both criteria 1 and 2.

3.4 Experimental Results

The number of events, E , in the LEM model is set to 25. The performance of the proposed

⁵TwiCal also groups event instances into event types such as "Sport" or "Politics" using LinkLDA which is not considered here.

Method	Tuple Evaluated	Precision	Recall	F-measure
Baseline	$\langle y, d, k \rangle$	75%	76.19%	75.59%
Proposed	$\langle y, d, l \rangle$	96%	80.95%	87.83%
Proposed	$\langle y, d, l, k \rangle$	92%	76.19%	83.35%

Table 1: Comparison of the performance of event extraction on the FSD dataset.

Method	Tuple Evaluated	Precision	Recall	F-measure
TW-NER	$\langle y, d, l \rangle$	88%	76.19%	80.35%
TW-NER	$\langle y, d, l, k \rangle$	84%	76.19%	79.90%
NW-NER	$\langle y, d, l \rangle$	96%	80.95%	87.83%
NW-NER	$\langle y, d, l, k \rangle$	92%	76.19%	83.35%

Table 2: Comparison of the performance of event extraction using different NER method.

framework is presented in Table 1. The baseline re-implemented here can only output 3-tuples $\langle y, d, k \rangle$ and we simply use the gold standard event trigger words to assign to k . Still, we observe that compared to the baseline approach, the performance of our proposed framework evaluated on the 4-tuple achieves nearly 17% improvement on precision. The overall improvement on F-measure is around 7.76%.

3.5 Impact of Named Entity Recognition

We experimented with two approaches for named entity recognition (NER) in preprocessing. One is to use the NER tool trained specifically on the Twitter data (Ritter et al., 2011), denoted as ‘‘TW-NER’’ in Table 2. The other uses the traditional Stanford NER to extract named entities from news articles published in the same period and then perform fuzzy matching to identify named entities from tweets. The latter method is denoted as ‘‘NW-NER’’ in Table 2. It can be observed from Table 2 that by using NW-NER, the performance of event extraction system is improved significantly by 7.5% and 3% respectively on F-measure when evaluated on 3-tuples (without keywords) or 4-tuples (with keywords).

3.6 Impact of the Number of Events E

We need to set the number of events E in the LEM model. Figure 3 shows the performance of event extraction versus different value of E . It can be observed that the performance of the proposed framework improves with the increase of the value of E until it reaches 25, which is close to the actual number of events in our data. If further increasing E , we notice more balanced precision/recall values and a relatively stable F-measure. This shows that our LEM model is less sensitive to the num-

ber of events E so long as E is set to a relatively larger value.

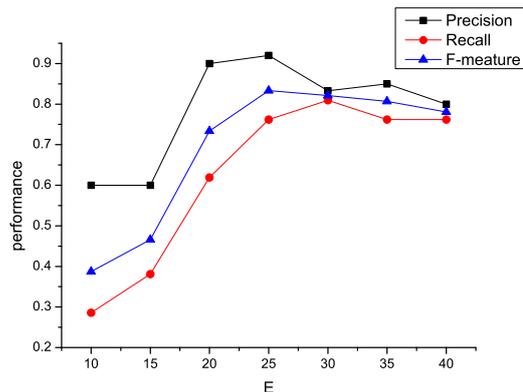


Figure 3: The performance of the proposed framework with different number of events E .

4 Conclusions and Future Work

In this paper we have proposed an unsupervised Bayesian model, called the Latent Event Model (LEM), to extract the structured representation of events from social media data. Instead of employing labeled corpora for training, the proposed model only requires the identification of named entities, locations and time expressions. After that, the model can automatically extract events which involving a named entity at certain time, location, and with event-related keywords based on the co-occurrence patterns of the event elements. Our proposed model has been evaluated on the FSD corpus. Experimental results show our proposed framework outperforms the state-of-the-art baseline by over 7% in F-measure. In future work, we plan to investigate inferring geolocations automatically from tweets. We also intend to study a better method to infer date more accurately from tweets and explore efficient ranking strategies to rank events extracted for a better presentation of results.

Acknowledgments

This work was funded by the National Natural Science Foundation of China (61103077), Ph.D. Programs Foundation of Ministry of Education of China for Young Faculties (20100092120031), Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, the Fundamental Research Funds for the Central Universities, and the UK’s EPSRC grant EP/L010690/1.

References

- Edward Benson, Aria Haghighi, and Regina Barzilay. 2011. Event discovery in social media feeds. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 389–398, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Angel X. Chang and Christopher D. Manning. 2012. SUTIME: A library for recognizing and normalizing time expressions. In *8th International Conference on Language Resources and Evaluation (LREC 2012)*.
- Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of ACL*.
- Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. In *Proceedings of the National Academy of Sciences 101 (Suppl. 1)*, page 5228C5235.
- Ralph Grishman, David Westbrook, and Adam Meyers. 2005. Nyu's english ace 2005 system description. In *ACE 05 Evaluation Workshop*.
- Frederik Hogenboom, Flavius Frasinca, Uzay Kaymak, and Franciska de Jong. 2011. An overview of event extraction from text. In *Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2011) at Tenth International Semantic Web Conference (ISWC2011)*, pages 48–57.
- Xiaohua Liu, Xiangyang Zhou, Zhongyang Fu, Furu Wei, and Ming Zhou. 2012. Extracting social events for tweets using a factor graph. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, pages 1692–1698.
- John Makhoul, Francis Kubala, Richard Schwartz, and Ralph Weischedel. 1999. Performance measures for information extraction. In *Proceedings of DARPA Broadcast News Workshop*.
- Inderjeet Mani and George Wilson. 2000. Robust temporal processing of news. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, ACL '00*, pages 69–76, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Sasa Petrovic, Miles Osborne, Richard McCreddie, Craig Macdonald, Iadh Ounis, and Luke Shrimpton. 2013. Can twitter replace newswire for breaking news? In *Proceedings of ICWSM'13*.
- J. Piskorski, H. Tanev, and P. Oezden Wennerberg. 2007. Extracting violent events from on-line news for ontology population. In *Business Information Systems*, pages 287–300.
- J. Piskorski, H. Tanev, M. Atkinson, and E. Van Der Goot. 2008. Cluster-centric approach to news event extraction. In *International Conference on New Trends in Multimedia and Network Information Systems*, pages 276–290.
- 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.
- Alan Ritter, Mausam, Oren Etzioni, and Sam Clark. 2012. Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12*, pages 1104–1112, New York, NY, USA. ACM.
- H. Tanev, J. Piskorski, and M. Atkinson. 2008. Real-time news event extraction for global crisis monitoring. In *13th International Conference on Applications of Natural Language to Information Systems (NLDB)*, pages 207–218.

Be Appropriate and Funny: Automatic Entity Morph Encoding

Boliang Zhang¹, Hongzhao Huang¹, Xiaoman Pan¹, Heng Ji¹, Kevin Knight²
Zhen Wen³, Yizhou Sun⁴, Jiawei Han⁵, Bulent Yener¹

¹Computer Science Department, Rensselaer Polytechnic Institute

²Information Sciences Institute, University of Southern California

³IBM T. J. Watson Research Center

⁴College of Computer and Information Science, Northeastern University

⁵Computer Science Department, University of Illinois at Urbana-Champaign

¹{zhangb8, huangh9, panx2, jih, yener}@rpi.edu, ²knight@isi.edu

³zhenwen@us.ibm.com, ⁴yzsun@ccs.neu.edu, ⁵hanj@illinois.edu

Abstract

Internet users are keen on creating different kinds of *morphs* to avoid censorship, express strong sentiment or humor. For example, in Chinese social media, users often use the entity morph “方便面 (*Instant Noodles*)” to refer to “周永康 (*Zhou Yongkang*)” because it shares one character “康 (*Kang*)” with the well-known brand of instant noodles “康师傅 (*Master Kang*)”. We developed a wide variety of novel approaches to automatically encode proper and interesting morphs, which can effectively pass decoding tests ¹.

1 Introduction

One of the most innovative linguistic forms in social media is *Information Morph* (Huang et al., 2013). Morph is a special case of alias to hide the original objects (e.g., sensitive entities and events) for different purposes, including avoiding censorship (Bamman et al., 2012; Chen et al., 2013), expressing strong sentiment, emotion or sarcasm, and making descriptions more vivid. Morphs are widely used in Chinese social media. Here is an example morphs: “由于瓜爹的事情，方便面与天线摊牌。(Because of *Gua Dad*’s issue, *Instant Noodles* faces down with *Antenna*.)”, where

- “瓜爹 (*Gua Dad*)” refers to “薄熙来 (*Bo Xilai*)” because it shares one character “瓜 (*Gua*)” with “薄瓜瓜 (*Bo Guagua*)” who is the son of “薄熙来 (*Bo Xilai*)”;
- “方便面 (*Instant Noodles*)” refers to “周永康 (*Zhou Yongkang*)” because it shares one character “康 (*kang*)” with the well-known instant noodles brand “康师傅 (*Master Kang*)”;

¹The morphing data set is available for research purposes: <http://nlp.cs.rpi.edu/data/morphencoding.tar.gz>

- “天线 (*Antenna*)” refers to “温家宝 (*Wen Jiabao*)” because it shares one character “宝 (*baby*)” with the famous children’s television series “天线宝宝 (*Teletubbies*)”;

In contrast with covert or subliminal channels studied extensively in cryptography and security, *Morphing* provides confidentiality against a weaker adversary which has to make a real time or near real time decision whether or not to block a morph within a time interval t . It will take longer than the duration t for a morph decoder to decide which encoding method is used and exactly how it is used; otherwise adversary can create a codebook and decode the morphs with a simple look up. We note that there are other distinct characteristics of morphs that make them different from cryptographic constructs: (1) Morphing can be considered as a way of using natural language to communicate confidential information without encryption. Most morphs are encoded based on semantic meaning and background knowledge instead of lexical changes, so they are closer to Jargon. (2) There can be multiple morphs for an entity. (3) The Shannon’s Maxim “the enemy knows the system” does not always hold. There is no common code-book or secret key between the sender and the receiver of a morph. (4) Social networks play an important role in creating morphs. One main purpose of encoding morphs is to disseminate them widely so they can become part of the new Internet language. Therefore morphs should be interesting, fun, intuitive and easy to remember. (5) Morphs rapidly evolve over time, as some morphs are discovered and blocked by censorship and newly created morphs emerge.

We propose a brand new and challenging research problem - can we automatically encode morphs for any given entity to help users communicate in an appropriate and fun way?

2 Approaches

2.1 Motivation from Human Approaches

Let's start from taking a close look at human's intentions and general methods to create morphs from a social cognitive perspective. In Table 1 and Table 2, we summarize 548 randomly selected morphs into different categories. In this paper we automate the first seven human approaches, without investigating the most challenging Method 8, which requires deep mining of rich background and tracking all events involving the entities.

2.2 M1: Phonetic Substitution

Given an entity name e , we obtain its phonetic transcription $pinyin(e)$. Similarly, for each unique term t extracted from Tsinghua Weibo dataset (Zhang et al., 2013) with one billion tweets from 1.8 million users from 8/28/2012 to 9/29/2012, we obtain $pinyin(t)$. According to the Chinese phonetic transcription articulation manner², the pairs (b, p) , (d, t) , (g, k) , (z, c) , (zh, ch) , (j, q) , (sh, r) , (x, h) , (l, n) , (c, ch) , (s, sh) and (z, zh) are mutually transformable.

If a part of $pinyin(e)$ and $pinyin(t)$ are identical or their initials are transformable, we substitute the part of e with t to form a new morph. For example, we can substitute the characters of “比尔 盖茨 (Bill Gates) [Bi Er Gai Ci]” with “鼻耳 (Nose and ear) [Bi Er]” and “盖子 (Lid) [Gai Zi]” to form new morph “鼻耳 盖子 (Nose and ear Lid) [Bi Er Gai Zi]”. We rank the candidates based on the following two criteria: (1) If the morph includes more negative words (based on a gazetteer including 11,729 negative words derived from HowNet (Dong and Dong, 1999), it's more humorous (Valitutti et al., 2013). (2) If the morph includes rarer terms with low frequency, it is more interesting (Petrovic and Matthews, 2013).

2.3 M2: Spelling Decomposition

Chinese characters are ideograms, hieroglyphs and mostly picture-based. It allows us to naturally construct a virtually infinite range of combinations from a finite set of basic units - radicals (Li and Zhou, 2007). Some of these radicals themselves are also characters. For a given entity name $e = c_1 \dots c_n$, if any character c_k can be decomposed into two radicals c_k^1 and c_k^2 which are both characters or can be converted into characters based on their pictograms (e.g., the radical “艹” can be

converted into“草” (grass)), we create a morph by replacing c_k with $c_k^1 c_k^2$ in e . Here we use a character to radical mapping table that includes 191 radicals (59 of them are characters) and 1328 common characters. For example, we create a morph “人呆罗 (Person Dumb Luo)” for “保罗 (Paul)” by decomposing “保 (Pau-)” into “人 (Person)” and “呆 (Dull)”. A natural alternative is to composing two character radicals in an entity name to form a morph. However, very few Chinese names include two characters with single radicals.

2.4 M3: Nickname Generation

We propose a simple method to create morphs by duplicating the last character of an entity's first name. For example, we create a morph “慕慕 (Mimi)” to refer to “杨 慕 (Yang Mi)”.

2.5 M4: Translation and Transliteration

Given an entity e , we search its English translation $EN(e)$ based on 94,015 name translation pairs (Ji et al., 2009). Then, if any name component in $EN(e)$ is a common English word, we search for its Chinese translation based on a 94,966 word translation pairs (Zens and Ney, 2004), and use the Chinese translation to replace the corresponding characters in e . For example, we create a morph “拉里 鸟儿 (Larry bird)” for “拉里 伯德 (Larry Bird)” by replacing the last name “伯德 (Bird)” with its Chinese translation “鸟儿 (bird)”.

2.6 M5: Semantic Interpretation

For each character c_k in the first name of a given entity name e , we search its semantic interpretation sentence from the Xinhua Chinese character dictionary including 20,894 entries³. If a word in the sentence contains c_k , we append the word with the last name of e to form a new morph. Similarly to M1, we prefer positive, negative or rare words. For example, we create a morph “薄 胡来 (Bo Mess)” for “薄熙来 (Bo Xi Lai)” because the semantic interpretation sentence for “来 (Lai)” includes a negative word “胡来 (Mess)”.

2.7 M6: Historical Figure Mapping

We collect a set of 38 famous historical figures including politicians, emperors, poets, generals, ministers and scholars from a website. For a given entity name e , we rank these candidates by applying the resolution approach as described in our previous work (Huang et al., 2013) to measure the similarity between an entity and a historic figure

²http://en.wikipedia.org/wiki/Pinyin#Initials_and_finals

³<http://xh.5156edu.com/>

Category	Frequency Distribution	Examples		
		Entity	Morph	Comment
(1) Avoid censorship	6.56%	薄熙来 (Bo Xilai)	B书记 (B Secretary)	“B” is the first letter of “Bo” and “Secretary” is the entity’s title.
(2) Express strong sentiment, sarcasm, emotion	15.77%	王勇平 (Wang Yongping)	奇迹哥 (Miracle Brother)	Sarcasm on the entity’s public speech: “It’s a miracle that the girl survived (from the 2011 train collision)”.
(3) Be humorous or make descriptions more vivid	25.91%	杨幂 (Yang Mi)	嫩牛五方 (Tender Beef Pentagon)	The entity’s face shape looks like the shape of famous KFC food “Tender Beef Pentagon”.
Mixture	25.32%	卡扎菲 (Gaddafi)	疯鸭上校 (Crazy Duck Colonel)	Sarcasm on Colonel Gaddafi’s violence.
Others	23.44%	蒋介石 (Chiang Kai-shek)	花生米 (Peanut)	Joseph Stilwell, a US general in China during World War II, called Chiang Kai-shek “花生米 (Peanut)” in his diary because of his stubbornness.

Table 1: Morph Examples Categorized based on Human Intentions

No.	Category	Frequency Distribution	Example		
			Entity	Morph	Comment
M1	Phonetic Substitution	12.77%	萨科齐 (Sarkozy)	傻客气 (Silly Polite)	The entity’s phonetic transcript “Sa Ke Qi” is similar to the morph’s “Sha Ke Qi”.
M2	Spelling Decomposition	0.73%	胡锦涛 (Hu Jintao)	古月 (Old Moon)	The entity’s last name is decomposed into the morph “古月 (Old Moon)”?
M3	Nickname Generation	12.41%	江泽民 (Jiang Zemin)	老江 (Old Jiang)	The morph is a conventional name for old people with last name “Jiang”.
M4	Translation & Transliteration	3.28%	布什 (Bush)	树丛 (shrub)	The morph is the Chinese translation of “bush”.
M5	Semantic Interpretation	20.26%	金日成 (Kim Il Sung)	金太阳 (Kim Sun)	The character “日” in the entity name means “太阳 (Sun)”.
M6	Historical Figure Mapping	3.83%	薄熙来 (Bo Xilai)	平西王 (Conquer West King)	The entity shares characteristics and political experiences similar to the morph.
M7	Characteristics Modeling	20.62%	金日成 (Kim Il Sung)	金胖子 (Kim Fat)	“胖子 (Fat)” describes “金日成 (Kim Il Sung)”’s appearance.
M8	Reputation and public perception	26.09%	奥巴马 (Obama)	观海 (Staring at the sea)	Barack Obama received a calligraphy “观海听涛 (Staring at sea and listening to surf)” as a present when he visited China.
			马景涛 (Ma Jingtao)	咆哮教主 (Roar Bishop)	In the films Ma Jingtao starred, he always used exaggerated roaring to express various emotions.
			马英九 (Ma Yingjiu)	马不统 (Ma Seccession)	The morph derives from Ma Yingjiu’s political position on cross-strait relations.

Table 2: Morph Examples Categorized based on Human Generation Methods

based on their semantic contexts. For example, this approach generates a morph “太祖 (the First Emperor)” for “毛泽东 (Mao Zedong)” who is the first chairman of P. R. China and “高祖 (the Second Emperor)” for “邓小平 (Deng Xiaoping)” who succeeded Mao.

2.8 M7: Characteristics Modeling

Finally, we propose a novel approach to automatically generate an entity’s characteristics using Google word2vec model (Mikolov et al., 2013). To make the vocabulary model as general as possible, we use all of the following large corpora that we have access to: Tsinghua Weibo dataset, Chinese Gigaword fifth edition⁴ which includes 10 million news documents, TAC-KBP 2009-2013 Source Corpora (McNamee and Dang, 2009; Ji et

al., 2010; Ji et al., 2011; Ji and Grishman, 2011) which include 3 million news and web documents, and DARPA BOLT program’s discussion forum corpora with 300k threads. Given an entity e , we compute the semantic relationship between e and each word from these corpora. We then rank the words by: (1) cosine similarity, (2) the same criteria as in section 2.6. Finally we append the top ranking word to the entity’s last name to obtain a new morph. Using this method, we are able to generate many vivid morphs such as “姚奇才 (Yao Wizard)” for “姚明 (Yao Ming)”.

3 Experiments

3.1 Data

We collected 1,553,347 tweets from Chinese Sina Weibo from May 1 to June 30, 2013. We extracted

⁴<http://catalog.ldc.upenn.edu/LDC2011T13>

187 human created morphs based on M1-M7 for 55 person entities. Our approach generated 382 new morphs in total.

3.2 Human Evaluation

We randomly asked 9 Chinese native speakers who regularly access Chinese social media and are not involved in this work to conduct evaluation independently. We designed the following three criteria based on Table 1:

- Perceivability: Who does this morph refer to? (i) Pretty sure, (ii) Not sure, and (iii) No clues.
- Funniness: How interesting is the morph? (i) Funny, (ii) Somewhat funny, and (iii) Not funny.
- Appropriateness: Does the morph describe the target entity appropriately? (i) Make sense, (ii) Make a little sense, and (iii) Make no sense.

The three choices of each criteria account for 100% (i), 50% (ii) and 0% (iii) satisfaction rate, respectively. If the assessor correctly predicts the target entity with the Perceivability measure, (s)he is asked to continue to answer the Funniness and Appropriateness questions; otherwise the Funniness and Appropriateness scores are 0. The human evaluation results are shown in Table 4. The Fleiss’s kappa coefficient among all the human assessors is 0.147 indicating slight agreement.

From Table 4 we can see that overall the system achieves 66% of the human performance with comparable stability as human. In particular, Method 4 based on translation and transliteration generates much more perceivable morphs than human because the system may search in a larger vocabulary. Interestingly, similar encouraging results - system outperforms human - have been observed by previous back-transliteration work (Knight and Graehl, 1998).

It’s also interesting to see that human assessors can only comprehend 76% of the human generated morphs because of the following reasons: (1) the morph is newly generated or it does not describe the characteristics of the target entity well; and (2) the target entity itself is not well known to human assessors who do not keep close track of news topics. In fact only 64 human generated morphs and 72 system generated morphs are perceivable by all human assessors.

For Method 2, the human created morphs are assessed as much more and funny than the system generated ones because human creators use this approach only if: (1). the radicals still reflect

the meaning of the character (e.g., “愁 (worry)” is decomposed into two radicals “心秋 (heart autumn)” instead of three “禾火心” (grain fire heart) because people tend to feel sad when the leaves fall in the autumn), (2). the morph reflects some characteristics of the entity (e.g., “江泽民 (Jiang Zemin)” has a morph “水工泽民 (Water Engineer Zemin)” because he gave many instructions on water conservancy construction); or (3). The morph becomes very vivid and funny (e.g., the morph “木子月月鸟 (Muji Yue Yue Bird)” for “李鹏” is assessed as very funny because “木子 (Muji)” looks like a Japanese name, “月月 (Yue Yue)” can also refer to a famous chubby woman, and “鸟人 (bird man)” is a bad word referring to bad people); or (4). The morph expresses strong sentiment or sarcasm; or (5) The morph is the name of another entity (e.g., the morph “古月 (Gu Yue)” for “胡锦涛 (Hu Jintao)” is also the name of a famous actor who often acts as Mao Zedong). The automatic approach didn’t explore these intelligent constraints and thus produced more boring morph. Moreover, sometimes human creators further exploit traditional Chinese characters, generalize or modify the decomposition results.

Table 3 presents some good (with average score above 80%) and bad (with average score below 20%) examples.

Good Examples		
Entity	Morph	Method
本拉登 (Osama bin Laden)	笨拉灯 (The silly turning off light)	M1
蒋介石 (Chiang Kai-shek)	草蒋介石 (Grass General Jie Shi)	M2
比尔盖茨 (Bill Gates)	票子盖茨 (Bill Gates)	M4
Bad Examples		
Entity	Morph	Method
科比 (Kobe)	胳膊 (Arm)	M1
梅德韦杰夫 (Medvedev)	梅德育 (Mei Virtue)	M5
林书豪 (Jeremy Lin)	老子 (Lao Tze)	M6

Table 3: System Generated Morph Examples

To understand whether users would adopt system generated morphs for their social media communication, we also ask the assessors to recite the morphs that they remember after the survey. Among all the morphs that they remember correctly, 20.4% are system generated morphs, which is encouraging.

3.3 Automatic Evaluation

Another important goal of morph encoding is to avoid censorship and freely communicate about

	M1		M2		M3		M4		M5		M6		M7		Overall	
	Human System		Human System		Human System		Human System		Human System		Human System		Human System		Human System	
# of morphs	17	124	4	21	10	54	9	28	64	87	9	18	74	50	187	382
Perceivability	75	76	95	86	94	81	61	71	87	59	66	5	77	34	76	67
Funniness	78	49	92	43	44	41	70	47	70	35	74	28	79	44	76	46
Appropriateness	71	51	89	59	81	43	75	49	76	36	78	18	82	38	79	43
Average	75	59	92	57	73	55	69	56	78	43	73	17	79	39	77	52
Standard Deviation	12.29	21.81	7.32	11.89	13.2	9.2	17.13	20.3	18.83	17.54	10.01	21.23	15.18	15.99	15.99	18.14

Table 4: Human Evaluation Satisfaction Rate (%)

certain entities. To evaluate how well the new morphs can pass censorship, we simulate the censorship using an automatic morph decoder consisted of a morph candidate identification system based on Support Vector Machines incorporating anomaly analysis and our morph resolution system (Huang et al., 2013). We use each system generated morph to replace its corresponding human-created morphs in Weibo tweets and obtain a new “morphed” data set. The morph decoder is then applied to it. We define *discovery rate* as the percentage of morphs identified by the decoder, and the ranking accuracy $Acc@k$ to evaluate the resolution performance. We conduct this decoding experiment on 247 system generated and 151 human generated *perceivable* morphs with perceivability scores $> 70\%$ from human evaluation.

Figure 1 shows that in general the decoder achieves lower discovery rate on system generated morphs than human generated ones, because the identification component in the decoder was trained based on human morph related features. This result is promising because it demonstrates that the system generated morphs contain new and unique characteristics which are unknown to the decoder. In contrast, from Figure 2 we can see that system generated morphs can be more easily resolved into the right target entities than human generated ones which are more implicit.

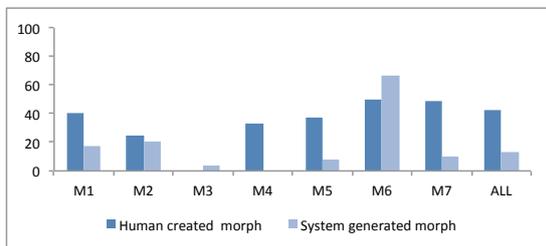


Figure 1: Discovery Rate (%)

4 Related Work

Some recent work attempted to map between Chinese formal words and informal words (Xia et al., 2005; Xia and Wong, 2006; Xia et al., 2006; Li

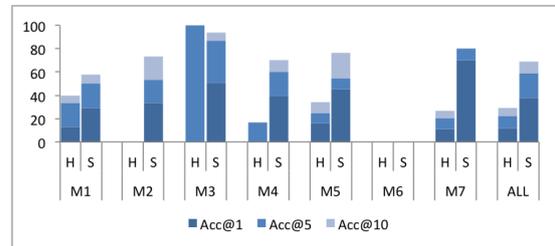


Figure 2: Resolution Acc@K Accuracy (%)

and Yarowsky, 2008; Wang et al., 2013; Wang and Kan, 2013). We incorporated the pronunciation, lexical and semantic similarity measurements proposed in these approaches. Some of our basic selection criteria are also similar to the constraints used in previous work on generating humors (Valitutti et al., 2013; Petrovic and Matthews, 2013).

5 Conclusions and Future Work

This paper proposed a new problem of encoding entity morphs and developed a wide variety of novel automatic approaches. In the future we will focus on improving the language-independent approaches based on historical figure mapping and culture and reputation modeling. In addition, we plan to extend our approaches to other types of information including sensitive events, satires and metaphors so that we can generate fable stories. We are also interested in tracking morphs over time to study the evolution of Internet language.

Acknowledgments

This work was supported by U.S. ARL No. W911NF-09-2-0053, DARPA No. FA8750-13-2-0041 and No. W911NF-12-C-0028, ARO No. W911NF-13-1-0193, NSF IIS-0953149, CNS-0931975, IIS-1017362, IIS-1320617, IIS-1354329, IBM, Google, DTRA, DHS and RPI. The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

- David Bamman, Brendan O'Connor, and Noah A. Smith. 2012. Censorship and deletion practices in Chinese social media. *First Monday*, 17(3).
- Le Chen, Chi Zhang, and Christo Wilson. 2013. Tweeting under pressure: analyzing trending topics and evolving word choice on sina weibo. In *Proceedings of the first ACM conference on Online social networks*, pages 89–100.
- Zhendong Dong and Qiang Dong. 1999. Hownet. In <http://www.keenage.com>.
- Hongzhao Huang, Zhen Wen, Dian Yu, Heng Ji, Yizhou Sun, Jiawei Han, and He Li. 2013. Resolving entity morphs in censored data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL2013)*.
- Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the Association for Computational Linguistics (ACL2011)*.
- Heng Ji, Ralph Grishman, Dayne Freitag, Matthias Blume, John Wang, Shahram Khadivi, Richard Zens, and Hermann Ney. 2009. Name extraction and translation for distillation. *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*.
- Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. 2010. Overview of the tac 2010 knowledge base population track. In *Text Analysis Conference (TAC) 2010*.
- Heng Ji, Ralph Grishman, and Hoa Trang Dang. 2011. Overview of the tac 2011 knowledge base population track. In *Proc. Text Analysis Conference (TAC) 2011*.
- Kevin Knight and Jonathan Graehl. 1998. Machine transliteration. *Computational Linguistics*, 24(4).
- Zhifei Li and David Yarowsky. 2008. Mining and modeling relations between formal and informal chinese phrases from web corpora. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP2008)*, pages 1031–1040.
- Jianyu Li and Jie Zhou. 2007. Chinese character structure analysis based on complex networks. *Physica A: Statistical Mechanics and its Applications*, 380:629–638.
- Paul McNamee and Hoa Trang Dang. 2009. Overview of the tac 2009 knowledge base population track. In *Proceedings of Text Analysis Conference (TAC2009)*.
- 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.
- Sasa Petrovic and David Matthews. 2013. Unsupervised joke generation from big data. In *Proceedings of the Association for Computational Linguistics (ACL2013)*.
- Alessandro Valitutti, Hannu Toivonen, Antoine Doucet, and Jukka M. Toivanen. 2013. "let everything turn well in your wife": Generation of adult humor using lexical constraints. In *Proceedings of the Association for Computational Linguistics (ACL2013)*.
- Aobo Wang and Min-Yen Kan. 2013. Mining informal language from chinese microtext: Joint word recognition and segmentation. In *Proceedings of the Association for Computational Linguistics (ACL2013)*.
- Aobo Wang, Min-Yen Kan, Daniel Andrade, Takashi Onishi, and Kai Ishikawa. 2013. Chinese informal word normalization: an experimental study. In *Proceedings of International Joint Conference on Natural Language Processing (IJCNLP2013)*.
- Yunqing Xia and Kam-Fai Wong. 2006. Anomaly detecting within dynamic chinese chat text. In *Proc. Workshop On New Text Wikis And Blogs And Other Dynamic Text Sources*.
- Yunqing Xia, Kam-Fai Wong, and Wei Gao. 2005. Nil is not nothing: Recognition of chinese network informal language expressions. In *4th SIGHAN Workshop on Chinese Language Processing at IJCNLP*, volume 5.
- Yunqing Xia, Kam-Fai Wong, and Wenjie Li. 2006. A phonetic-based approach to chinese chat text normalization. In *Proceedings of COLING-ACL2006*, pages 993–1000.
- Richard Zens and Hermann Ney. 2004. Improvements in phrase-based statistical machine translation. In *Proceedings of HLT-NAACL2004*.
- Jing Zhang, Biao Liu, Jie Tang, Ting Chen, and Juanzi Li. 2013. Social influence locality for modeling retweeting behaviors. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13)*, pages 2761–2767.

Applying Grammar Induction to Text Mining

Andrew Salway

Uni Research Computing
Thormøhlensgt. 55
N-5008 Bergen
Norway

andrew.salway@uni.no

Samia Touileb

Information Science and Media Studies
University of Bergen
N-5020 Bergen
Norway

samia.touileb@gmail.com

Abstract

We report the first steps of a novel investigation into how a grammar induction algorithm can be modified and used to identify salient information structures in a corpus. The information structures are to be used as representations of semantic content for text mining purposes. We modify the learning regime of the ADIOS algorithm (Solan et al., 2005) so that text is presented as increasingly large snippets around key terms, and instances of selected structures are substituted with common identifiers in the input for subsequent iterations. The technique is applied to 1.4m blog posts about climate change which mention diverse topics and reflect multiple perspectives and different points of view. Observation of the resulting information structures suggests that they could be useful as representations of semantic content. Preliminary analysis shows that our modifications had a beneficial effect for inducing more useful structures.

1 Introduction

There is an obvious need for text mining techniques to deal with large volumes of very diverse material, especially since the advent of social media and user-generated content which includes dynamic discussions of wide-ranging and controversial topics.

In order to be portable across domains, text genres and languages, current techniques tend to treat texts as bags of words when analyzing semantic content, e.g. for keyword-based retrieval, summarization with word clouds, and topic modelling. Such techniques capture the general “aboutness” of texts, but they do little to elucidate the actual statements that are made about key terms in the material. More structured and deeper semantic representations can be

generated by information extraction systems for relatively restricted text genres and domains, but even then they are costly to port.

We see one particular area of application in elucidating the semantic content of social media debates about controversial topics, like climate change, both for casual users, and for social scientists studying online discourses. The complex, diverse and dynamic nature of the text content in such material presents a significant challenge for elucidating semantics. On the one hand, keywords alone will not convey what is said about important concepts, nor different points of view. On the other hand, modelling the semantics for information extraction purposes does not seem feasible given the breadth and diversity of the material.

Thus, we are motivated to develop a portable technique that generates representations of semantic content that are richer than keywords, and that can be applied to broad domains. Specifically, we seek to extract important information structures from an unannotated corpus comprising texts of the same genre and relating to the same domain.

Rather than using language-specific or domain-specific resources, we assume that important information structures in such a corpus will be reflected by patterning in the surface form of texts, such that they can be identified automatically through a distributional analysis (Section 2). Our approach is to induce information structures from an unannotated corpus by modifying and applying the ADIOS grammar induction algorithm (Solan et al., 2005): the modifications serve to focus the algorithm on what is typically written about key-terms (Section 3). To date we have implemented the approach to process 1.4m English-language blog posts about climate change: proper evaluation is ongoing but we are able to show

examples of the semantic representations generated, discuss how they elucidate semantic content, and suggest how they might be used for various NLP tasks (Section 4). In closing, we make tentative conclusions and describe ongoing work (Section 5).

2 Background

Harris (1954; 1988) demonstrated how linguistic units and structures can be identified (manually) through a distributional analysis of partially aligned sentential contexts. His work suggests that it should be possible to induce syntactic descriptions from samples of unannotated text.

An early attempt to apply this thinking to computational linguistics was made by Lamb (1961) who described procedures for identifying “H-groups” and “V-groups”. An H-group is a horizontal grouping of items (words and groups) that tend to appear sequentially, cf. a syntagmatic linguistic unit. A V-group is a vertical grouping of items that occur in similar linguistic contexts in a corpus, cf. a paradigmatic linguistic unit. As a toy example, take the H-group `^(the (woman | man) went to the (pub | shop | park))'`, with V-groups `^(woman | man)'` and `^(pub | shop | park)'`.

In more recent times, Harris’ insights have become a cornerstone for some of the work in the field of grammatical inference, where researchers attempt to induce grammatical structures from raw text, e.g. ADIOS (Solan et al., 2005). In this field the emphasis is on generating complete grammatical descriptions for text corpora in order to understand the processes of language learning, rather than text mining; see D’Ulizia et al. (2011) for a review.

The unsupervised ADIOS algorithm recursively induces hierarchically structured patterns from sequential data, e.g. sequences of words in unannotated text, using statistical information in the sequential data. Each sequence (sentence) is loaded onto a directed pseudograph with one vertex for each vocabulary item: this means that partially aligned sequences share sub-paths across the graph.

In each iteration, the most significant pattern is identified with a statistical criterion that favors frequent sequences that occur in a variety of contexts. Then, the algorithm looks for possible equivalence classes within the context of the pattern, i.e. it identifies positions in the pattern that could be filled by different items and forms an equivalence class with those items. At the end

of the iteration, the new pattern and equivalence class become vocabulary items in the graph, so that they can become part of further patterns and equivalence classes, and hence hierarchical structures are formed. For us, the terms “pattern” and “equivalence class” equate to the previously mentioned “H-group” and “V-group”: we prefer the simplicity and literalness of these terms and use them henceforth.

3 Approach

For text mining purposes we do not see the need to induce a complete grammar for the corpus that we are mining. Rather, we are struck by Harris’ further observation that the linguistic structures derived from a distributional analysis may reflect information structures, especially in the “sublanguages” of specialist domains (Harris, 1988). Thus, we propose to use a grammar induction algorithm to identify the most salient information structures in a corpus and take these as representations of important semantic content.

ADIOS has been evaluated on an interesting range of text corpora, and other kinds of sequential data. However, to the best of our knowledge, it has not been shown to successfully process a corpus with the scale and diversity of material that we envisage, e.g. 1.4m blog posts relating to climate change. This, along with our objective of identifying salient information structures rather than a complete grammatical description, led us to modify the learning regime to ADIOS. In the rest of this section we explain the modifications: please see 4.2.1 for a detailed description of how they were implemented.

To address the large scale and complexity of language use in social media, we modify the way in which text is presented to ADIOS by focusing separately on text around key terms of interest, rather than processing all sentences en masse. Our thinking here is in part influenced by the theory of local grammar (Gross, 1997), i.e. the idea that language is best described with word classes that are specific to local contexts, rather than general across the language.

Firstly, for each key term, we present only text snippets that contain that term: we expect there to be more salient patterning in snippets around a single key term because of repetition in the kinds of things written about it. Secondly, blog posts contain long and complex sentences so we process the clause containing a key term, and ignore the rest of the sentence. Thirdly, since we expect the key term to form more significant

units with words in its close proximity, we present the clauses in increasingly large snippets around the key term.

A further modification targets the most frequent and meaningful structures. After each iteration in which H-groups and V-groups are induced, the most frequent H-groups are filtered to remove any containing large V-groups which are likely to be more semantically nebulous. Instances of the selected H-groups are replaced with common identifiers in the input file so that patterning around them is more explicit in subsequent iterations.

4 Implementation

Here we report our first attempt to apply grammar induction to text mining. We chose to work with a corpus of blogs relating to climate change because they provide a challenging scenario with complex semantics, in which diverse topics – causes, effects, solutions, etc. – are discussed from multiple perspectives – scientific, political, personal, etc. – and with different beliefs (section 4.1).

We describe how we modified the learning regime of the ADIOS algorithm in order to induce H-groups and V-groups from an unannotated corpus (4.2.1). At this stage in our work, our focus is on observing the kinds of information structures that can be identified in this way, and in considering their potential applications as representations of semantic content (4.2.2). We also analyzed how results were affected by our modifications, i.e. the use of incrementally bigger snippets rather than complete clauses, and the iterative selection and substitution of frequent H-groups (4.2.3).

4.1 Input data

We used a corpus of about 1.4m unannotated English-language blog posts from 3,000 blogs related to climate change (Salway et al., 2013). Based on the relative frequency of words compared with a general language corpus, and the use of n-grams, we identified a set of domain key terms, e.g. ‘climate change’, ‘greenhouse gases’, ‘carbon tax’, ‘sea levels’. From these we selected 17, with a mix of high (10,000’s), medium (1,000’s) and low (100’s) frequencies.

For each key term we crudely extracted every clause it occurred in by taking a clause to be a sequence of words between punctuation. Pre-processing involved conversion to lower case, joining the words of key terms to make single

items, e.g. ‘greenhouse_gases’, and substituting ‘dddd’ with ‘YEAR’, and other digit sequences with ‘NUMBER’: these changes all serve to make patterning more explicit.

Then, from the clauses for each key term, snippets of varying sizes were created. A snippet file for a key term is defined by (min-max) where there must be at least min words to one side of the key term, and no more than max words either side. Sets of snippet files were created for three different increment values: $i = 2$ (0-2, 3-4, 5-6, 7-8, 9-10, 11-12); $i = 3$ (0-3, 4-6, 7-9, 10-12); and, $i = 4$ (0-4, 5-8, 9-12).

4.2 Modifying the ADIOS learning regime

4.2.1 Method

In Section 3 we explained the rationale for our modifications to the ADIOS learning regime. They are detailed in steps 1 and 3-5 below.

For one key term and one increment value:

- (1) INITIALIZE. Set the current input file to be the first snippet file for the key term and increment value, i.e. the smallest snippets.
- (2) INDUCE CANDIDATE H-GROUPS AND V-GROUPS. Run the ADIOS algorithm over the current input file with default parameter values, except $E=0.9$ (cf. Solan et al. 2005).
- (3) SELECTION. Filter the 5 most frequent H-groups to keep those that meet the following criterion: if the H-group contains a V-group then the V-group must contain < 6 elements. If none of the 5 most frequent H-groups remain then go to (5).
- (4) SUBSTITUTION. For each selected H-group, replace all instances of it in the current input file with a common identifier. Iterate 10 times from (2).
- (5) TRANSITION. Until the final snippet file is reached, set the current input file to be the next largest snippet file and substitute identifiers for the instances of all H-groups selected so far. Go to (2).

This process was executed for 17 key terms, with three increment values ($i = 2, 3, 4$). For further comparison, for each key term it was executed with complete clauses (ten iterations with selection and substitution) and with complete clauses (one iteration).

1.	((to (combat fight)) (to (battle slow minimise mitigate tackle))) climate_change)
2.	(climate_change (summit adaptation talks meetings convention))
3.	((greenhouse gases) emissions gases (carbon emissions) pollution) blamed ((for to) global_warming)
4.	((cause causes) (of global_warming))
5.	((dangers signs effect consequences perils) (of global_warming))
6.	(to (confuse mislead educate) the public) // from global_warming snippets
7.	((anthropogenic manmade (man made)) global_warming)
8.	((would should to must) (control reduce regulate regulating release) greenhouse_gases)
9.	((source emitter emitters producers) of greenhouse_gases)
10.	(the (effects impact) ((under of) ((a its the) carbon_tax)))
11.	(a (modest \$_NUMBER a tonne global simple) carbon_tax)
12.	((will would to) (push raise elevate) (sea_levels (around by)))
13.	((due to) (caused by)) ((climate change) (global warming)) //from sea_levels snippets
14.	((the global some sophisticated complex) climate_models) (hint show indicate) that)

Table 1. A small selection of H-groups induced from snippets for a variety of key terms (in bold).

4.2.2 Results and potential applications

Table 1 presents a small selection of 14 H-groups that were induced from snippets with various key terms and increment values. Here, H-groups and V-groups are bracketed and nested. The elements of H-groups are separated by white space and the elements of V-groups are separated by '|'. Recall that the induction process selects frequent H-groups which, based on our assumptions, should reflect important semantic content.

This output would benefit from some post-processing, which is part of ongoing work. For example, in 1 there are two V-groups containing verbs that would be more elegantly expressed as a single V-group. There are also H-groups in which not all V-group alternatives make sense with the rest of the containing H-group due to over-generalization, e.g. 'to' in '...blamed ((for|to) global warming)' in 3. Despite these issues, some interesting and potentially useful structures are induced.

Some H-groups, we assume those resulting from the most stylized use of language in blogs, could perhaps be taken as the basis for information extraction templates, e.g. 11 where '\$_NUMBER' is a slot for different amounts of tax, and 12 which captures various ways in which predictions about the amount of sea level rise can be written.

Other H-groups highlight some of the things typically written about key terms by grouping together different expressions of canonical

statements, e.g. 3, 8 and 13. These could be used as a basis for summarizing the most important points of a topic, i.e. by taking 10,000's sentences and reducing them to 10's H-groups.

For broad topics it is desirable to perform finer-grained text classification and retrieval. The induction of H-groups such as 4 and 5 helps to identify different facets of a topic. In this case, the H-groups flag the causes of global warming and the effects of global warming as sub-topics, and show different ways in which they may be expressed.

The alternation in V-groups contained by H-groups may reflect different beliefs and opinions which could be used for text classification and opinion mining. In 14, the V-group 'hint|show|indicate' reflects different degrees of confidence that bloggers have in climate models. In 6, the alternatives in 'confuse|mislead|educate' reflect positive and negative views about public communication in the climate debate.

Semantically related terms, such as those captured in 1 and 5, have very different connotations and as such reflect different beliefs: consider the difference between someone writing about the 'effect of global warming' and the 'perils of global warming'. In other cases, alternation reflects different ways to say the same thing, e.g. the more or less synonymous terms that are captured in 2, 7 and 9 which would be useful for query expansion.

Key Term	Clauses	Number of different H-groups and <i>total instances</i>									
		i=2		i=3		i=4		clauses-10		clauses-1	
climate change	48241	198	47000	105	52745	86	57799	8	31611	698	123531
global warming	27582	191	25998	155	30001	104	31850	40	32315	397	57388
greenhouse gases	20345	174	30148	136	34009	94	33846	28	25213	552	65167
carbon tax	7751	106	6727	84	8341	80	9859	36	11393	128	14988
sea levels	6448	138	8322	121	10246	118	11020	55	12090	240	16752
climate models	6276	98	5041	91	6020	74	6399	26	6061	142	11058
emissions trading scheme	2989	86	2243	65	3802	68	3140	50	7680	96	8118

Table 2. Numbers of different H-groups and total instances generated from different input data.

4.2.3 The effects of our modifications

The numbers of H-groups generated by different executions of the induction process for each key term are shown in Table 2, i.e. three executions using snippets with different values of i , and two executions using clauses for comparison (cf. 4.2.1). The 10 omitted key terms (less than 1,000 clauses each) generated less than 25 H-groups for each value of i .

The high frequencies for clauses-1 are because no selection of H-groups took place, i.e. we simply take the normal ADIOS output. Based on our own inspections, some potentially useful H-groups were found in this output but, compared with other outputs, it was more common to see H-groups with large and semantically nebulous V-groups. This observation supports the iterative selection and substitution of H-groups with a limit on the size of V-groups. We also looked at the average number of V-groups in H-groups for each execution, as a way to compare the amount of structure in H-groups. This number was consistently lowest in results for clauses-1 which further supports our modifications.

A few potentially useful H-groups were observed in results for clauses-10, for which selection and substitution were applied. However the low numbers of different H-groups compared with all values of i suggests that it is better to use snippets as input rather than clauses.

The way in which the ratio of different H-groups and total instances varies for values of i suggests that starting with larger snippets ($i=4$) results in fewer H-groups but that these will capture more instances, i.e. they are more general. Whilst the H-groups for clauses-10 have many instances these tend not to capture useful patterning, i.e. they tended to describe combinations of key terms and function words.

5 Closing Remarks

At this stage in the research any conclusions must be tentative. However, it seems to us that

the use of grammar induction to elucidate semantic content for text mining purposes shows promise. The H-groups shown in Table 1 provide richer semantic descriptions of the domain than keywords do, and we noted potential applications for high-level summarization of a whole corpus, the creation of information extraction templates and finer-grained text classification and retrieval. Importantly, the technique for generating H-groups would not require adaptation for use on a different corpus. The analysis in 4.2.3 suggests that the modifications that we made to the ADIOS learning regime had a beneficial effect.

Without a thorough evaluation we cannot make strong claims. In particular, we have little sense of the technique's recall, i.e. we do not know what information structures it missed. That said, it might be argued that since we expect the technique to be consistent in identifying patterning in the surface form of texts then its success will depend on the extent to which key terms are written about in consistent ways. This will of course vary between text genres and domains. Work has started on another corpus with more restricted language use and richer structuring was induced (Salway et al. 2014).

In other ongoing work we are looking more into the effects of the various parameters of ADIOS, and the necessity for our modifications. We are also seeking a deeper understanding of how the statistical information exploited by ADIOS relates to that which is captured by n -gram language models to describe sequences of words (cf. H-groups), and by established techniques to form semantic classes based on shared linguistic contexts (cf. V-groups).

Acknowledgments

We are very grateful to Zach Solan for providing an implementation of the ADIOS algorithm, and to Knut Hofland and Lubos Steskal for their roles in creating the NTAP blog corpus. This research was supported by a grant from The Research Council of Norway's VERDIKT program.

References

- Arianna D'Ulizia, Fernando Ferri and Patrizia Grifoni. 2011. A survey of grammatical inference methods for natural language learning. *Artificial Intelligence Review* 36(1):1-27.
- Maurice Gross. 1997. The Construction of Local Grammars. In: E. Roche and Y. Schabes (eds.), *Finite-State Language Processing*. The MIT Press, Cambridge MA: 329-354.
- Zellig Harris. 1954. Distributional Structure. *Word* 10(2/3):146-162.
- Zellig Harris. 1988. *Language and Information*. Columbia University Press, New York.
- Sydney Lamb. 1961. On the Mechanization of Syntactic Analysis. *Int. Conf. Machine Translation of Languages and Applied Language Analysis*.
- Andrew Salway, Knut Hofland and Samia Touileb. 2013. Applying Corpus Techniques to Climate Change Blogs. *Procs. Corpus Linguistics 2013*, Lancaster University.
- Andrew Salway, Samia Touileb and Endre Tivinnereim. 2014. Inducing Information Structures for Data-driven Text Analysis. To appear in: *Procs. ACL Workshop on Language Technologies and Computational Social Science*.
- Zach Solan, David Horn, Eytan Ruppin, and Shimon Edelman. 2005. Unsupervised learning of natural languages. *Procs. of the National Academy of Sciences* 102(33):11629-11634.

Semantic Consistency: A Local Subspace Based Method for Distant Supervised Relation Extraction

Xianpei Han and Le Sun

State Key Laboratory of Computer Science
Institute of Software, Chinese Academy of Sciences
HaiDian District, Beijing, China.

{xianpei, sunle}@nfs.iscas.ac.cn

Abstract

One fundamental problem of distant supervision is the noisy training corpus problem. In this paper, we propose a new distant supervision method, called *Semantic Consistency*, which can identify reliable instances from noisy instances by inspecting whether an instance is located in a semantically consistent region. Specifically, we propose a *semantic consistency* model, which first models the local subspace around an instance as a sparse linear combination of training instances, then estimate the semantic consistency by exploiting the characteristics of the local subspace. Experimental results verified the effectiveness of our method.

1 Introduction

Relation extraction aims to identify and categorize relations between pairs of entities in text. Due to the time-consuming annotation process, one critical challenge of relation extraction is the lack of training data. To address this limitation, a promising approach is *distant supervision (DS)*, which can automatically gather labeled data by heuristically aligning entities in text with those in a knowledge base (Mintz et al., 2009). The underlying assumption of distant supervision is that every sentence that mentions two entities is likely to express their relation in a knowledge base.

Relation Instance	Label
S1: <i>Jobs was the founder of Apple</i>	Founder-of, CEO-of
S2: <i>Jobs joins Apple</i>	Founder-of, CEO-of

Figure 1. Labeled instances by distant supervision, using relations *CEO-of*(*Steve Jobs, Apple Inc.*) and *Founder-of*(*Steve Jobs, Apple Inc.*)

The distant supervision assumption, unfortunately, can often fail and result in a noisy training corpus. For example, in Figure 1 DS assumption will wrongly label S1 as a *CEO-of* instance and S2

as instance of *Founder-of* and *CEO-of*. The noisy training corpus in turn will lead to noisy extractions that hurt extraction accuracy (Riedel et al., 2010).

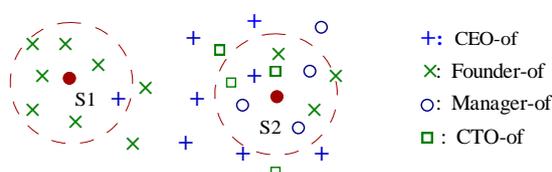


Figure 2. The regions the two instances in Figure 1 located, where: 1) S1 locates in a semantically consistent region; and 2) S2 locates in a semantically inconsistent region

To resolve the noisy training corpus problem, this paper proposes a new distant supervision method, called *Semantic Consistency*, which can effectively identify reliable instances from noisy instances by inspecting whether an instance is located in a semantically consistent region. Figure 2 shows two intuitive examples. We can see that, semantic consistency is an effective way to identify reliable instances. For example, in Figure 2 S1 is highly likely a reliable *Founder-of* instance because its neighbors are highly semantically consistent, i.e., most of them express the same relation type – *Founder-of*. On contrast S2 is highly likely a noisy instance because its neighbors are semantically inconsistent, i.e., they have a diverse relation types. The problem now is how to model the semantic consistency around an instance.

To model the semantic consistency, this paper proposes a local subspace based method. Specifically, given sufficient training instances, our method first models each relation type as a linear subspace spanned by its training instances. Then, the local subspace around an instance is modeled and characterized by seeking the sparsest linear combination of training instances which can reconstruct the instance. Finally, we estimate the semantic consistency of an instance by exploiting the characteristics of its local subspace.

This paper is organized as follows. Section 2 reviews related work. Section 3 describes the proposed method. Section 4 presents the experiments. Finally Section 5 concludes this paper.

2 Related Work

This section briefly reviews the related work. Craven and Kumlien (1999), Wu et al. (2007) and Mintz et al.(2009) were several pioneer work of distant supervision. One main problem of DS assumption is that it often will lead to false positives in training data. To resolve this problem, Bunescu and Mooney (2007), Riedel et al. (2010) and Yao et al. (2010) relaxed the DS assumption to the *at-least-one* assumption and employed multi-instance learning techniques to identify wrongly labeled instances. Takamatsu et al. (2012) proposed a generative model to eliminate noisy instances.

Another research issue of distant supervision is that a pair of entities may participate in more than one relation. To resolve this problem, Hoffmann et al. (2010) proposed a method which can combine a sentence-level model with a corpus-level model to resolve the multi-label problem. Surdeanu et al. (2012) proposed a multi-instance multi-label learning approach which can jointly model all instances of an entity pair and all their labels. Several other research issues also have been addressed. Xu et al. (2013), Min et al. (2013) and Zhang et al. (2013) try to resolve the false negative problem raised by the incomplete knowledge base problem. Hoffmann et al. (2010) and Zhang et al. (2010) try to improve the extraction precision by learning a dynamic lexicon.

3 The Semantic Consistency Model for Relation Extraction

In this section, we describe our semantic consistency model for relation extraction. We first model the subspaces of all relation types in the original feature space, then model and characterize the local subspace around an instance, finally estimate the semantic consistency of an instance and exploit it for relation extraction.

3.1 Testing Instance as a Linear Combination of Training Instances

In this paper, we assume that there exist k distinct relation types of interest and each relation type is represented with an integer index from 1 to k . For i th relation type, we assume that totally n_i training instances $V_i = \{\mathbf{v}_{i,1}, \mathbf{v}_{i,2}, \dots, \mathbf{v}_{i,n_i}\}$ have been collected using DS assumption. And each instance is represented as a weighted feature vector, such

as the features used in (Mintz, 2009) or (Surdeanu et al., 2012), with each feature is TFIDF weighted by taking each instance as an individual document.

To model the subspace of i th relation type in the original feature space, a variety of models have been proposed to discover the underlying patterns of V_i . In this paper, we make a simple and effective assumption that *the instances of a single relation type can be represented as the linear combination of other instances of the same relation type*. This assumption is well motivated in relation extraction, because although there is nearly unlimited ways to express a specific relation, in many cases basic principles of economy of expression and/or conventions of genre will ensure that certain systematic ways will be used to express a specific relation (Wang et al., 2012). For example, as shown in (Hearst, 1992), the *IS-A* relation is usually expressed using several regular patterns, such as “*such NP as {NP ,} * {(or | and)} NP*” and “*NP { , NP} * { , } or other NP*”.

Based on the above assumption, we hold many instances for each relation type and directly use these instances to model the subspace of a relation type. Specifically, we represent an instance \mathbf{y} of i th type as the linear combination of training instances associated with i th type:

$$\mathbf{y} = \alpha_{i,1}\mathbf{v}_{i,1} + \alpha_{i,2}\mathbf{v}_{i,2} + \dots + \alpha_{i,n_i}\mathbf{v}_{i,n_i} \quad (1)$$

for some scalars $\alpha_{i,j}$, with $j = 1, 2, \dots, n_i$. For example, we can represent the *CEO-of* instance “*Jobs was the CEO of Apple*” as the following linear combination of *CEO-of* instances:

- 0.8: *Steve Ballmer is the CEO of Microsoft*
- 0.2: *Rometty was served as the CEO of IBM*

For simplicity, we arrange the given n_i training instances of i th relation type as columns of a matrix $A_i = [\mathbf{v}_{i,1}, \mathbf{v}_{i,2}, \dots, \mathbf{v}_{i,n_i}]$, then we can write the matrix form of Formula 1 as:

$$\mathbf{y} = \mathbf{A}_i \mathbf{x}_i \quad (2)$$

where $\mathbf{x}_i = [\alpha_{i,1}, \dots, \alpha_{i,n_i}]$ is the coefficient vector. In this way, the subspace of a relation type is the linear subspace spanned by its training instances, and if we can find a valid \mathbf{x}_i , we can explain \mathbf{y} as a valid instance of i th relation type.

3.2 Local Subspace Modeling via Sparse Representation

Based on the above model, the local subspace of an instance is modeled as the linear combination of training instances which can reconstruct the instance. Specifically, to model the local subspace, we first concatenate the n training instances of all k relation types:

$$A = [A_1, A_2, \dots, A_k]$$

Then the local subspace around \mathbf{y} is modeled by seeking the solution of the following formula:

$$\mathbf{y} = \mathbf{A}\mathbf{x} \quad (3)$$

However, because of the redundancy of training instances, Formula 3 usually has more than one solution. In this paper, following the idea in (Wright et al., 2009) for robust face recognition, we use the sparsest solution (i.e., how to reconstruct an instance using minimal training instances), which have been shown is both discriminant and robust to noisiness. Concretely, we seek the sparse linear combination of training instances to reconstruct \mathbf{y} by solving:

$$(l^1): \mathbf{x}^* = \arg \min \|\mathbf{x}\|_1 \text{ s.t. } \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2 \leq \epsilon \quad (4)$$

where $\mathbf{x} = [\alpha_{1,1}, \dots, \alpha_{1,n_1}, \dots, \alpha_{i,1}, \alpha_{i,2}, \dots, \alpha_{i,n_i}, \dots]$ is a coefficient vector which identifies the spanning instances of \mathbf{y} 's local subspace, i.e., the instances whose $\alpha_{i,j} \neq 0$. In practice, the training corpus may be too large to direct solve Formula 4. Therefore, this paper uses the K-Nearest-Neighbors (KNN) of \mathbf{y} (1000 nearest neighbors in this paper) to construct the training instance matrix \mathbf{A} for each \mathbf{y} , and KNN can be searched very efficiently using specialized algorithms such as the LSH functions in (Andoni & Indyk, 2006).

Through the above semantic decomposition, we can see that, the entries of \mathbf{x} can encode the underlying semantic information of instance \mathbf{y} . For i th relation type, let $\delta_i(\mathbf{x})$ be a new vector whose only nonzero entries are the entries in \mathbf{x} that are associated with i th relation type, then we can compute the semantic component corresponding to i th relation type as $\mathbf{y}_i = \mathbf{A}\delta_i(\mathbf{x})$. In this way a testing instance \mathbf{y} will be decomposed into k semantic components, with each component corresponds to one relation type (with an additional noise component ϵ):

$$\mathbf{y} = \mathbf{y}_1 + \dots + \mathbf{y}_i + \dots + \mathbf{y}_k + \epsilon \quad (5)$$

$$\begin{aligned} \text{S1} &= 0.8 \times \begin{bmatrix} \text{was} \\ \text{co-founder} \\ \text{of} \\ \dots \end{bmatrix} + 0.2 \times \begin{bmatrix} \text{Jobs} \\ \text{Apple} \\ \text{the} \\ \dots \end{bmatrix} \\ &\quad \text{Founder-of} \qquad \qquad \text{noise} \\ \text{S2} &= 0.1 \begin{bmatrix} \text{join} \\ \dots \end{bmatrix} + 0.1 \begin{bmatrix} \text{join} \\ \dots \end{bmatrix} + 0.1 \begin{bmatrix} \text{join} \\ \dots \end{bmatrix} + \dots \\ &\quad \text{Founder-of} \quad \text{CEO-of} \quad \text{CTO-of} \end{aligned}$$

Figure 3. The semantic decomposition of the two instances in Figure 1

Figure 3 shows an example of semantic decomposition. We can see that, the semantic decomposition can effectively summarize the semantic

consistency information of \mathbf{y} 's local subspace: if the instances around an instance have diverse relation types (S2 for example), its information will be scattered on many different semantic components. On contrast if the instances around an instance have consistent relation types (S1 for example), most of its information will concentrate on the corresponding relation type.

3.3 Semantic Consistency based Relation Extraction

This section describes how to estimate and exploit the semantic consistency for relation extraction. Specifically, given \mathbf{y} 's semantic decomposition:

$$\mathbf{y} = \mathbf{y}_1 + \dots + \mathbf{y}_i + \dots + \mathbf{y}_k + \epsilon$$

we observe that if instance \mathbf{y} locates at a semantic consistent region, then all its information will concentrate on a specific component \mathbf{y}_i , with all other components equal to zero vector $\mathbf{0}$. However, modeling errors, expression ambiguity and noisy features will lead to small nonzero components. Based on the above discussion, we define the semantic consistency of an instance as the semantic concentration degree of its decomposition:

Definition 1(Semantic Consistency). For an instance \mathbf{y} , its semantic consistency with i th relation type is:

$$\text{Consistency}(\mathbf{y}, i) = \frac{\|\mathbf{y}_i\|_2}{\sum_i \|\mathbf{y}_i\|_2 + \|\epsilon\|_2}$$

where $\text{Consistency}(\mathbf{y}, i) \in [0, 1]$ and will be 1.0 if all information of \mathbf{y} is consistent with i th relation type; on contrast it will be 0 if no information in \mathbf{y} is consistent with i th relation type.

Semantic Consistency based Relation Extraction. To get accurate extractions, we determine the relation type of \mathbf{y} based on both: 1) How much information in \mathbf{y} is related to i th type; and 2) its semantic consistency score with i th type, i.e., whether \mathbf{y} is a reliable instance of i th type.

To measure how much information in \mathbf{y} is related to i th relation type, we compute the proportion of common information between \mathbf{y} and \mathbf{y}_i :

$$\text{sim}(\mathbf{y}, \mathbf{y}_i) = \frac{\mathbf{y} \cdot \mathbf{y}_i}{\mathbf{y} \cdot \mathbf{y}} \quad (6)$$

Then the likelihood for a testing instance \mathbf{y} expressing i th relation type is scored by summarizing both its information and semantic consistency:

$$\text{rel}(\mathbf{y}, i) = \text{sim}(\mathbf{y}, \mathbf{y}_i) \times \text{Consistency}(\mathbf{y}, i)$$

and \mathbf{y} will be classified into i th relation type if its likelihood is larger than a threshold:

$$\text{rel}(\mathbf{y}, i) \geq \tau_i \quad (7)$$

where τ_i is a relation type specific threshold learned from training dataset.

Multi-Instance Evidence Combination. It is often that an entity pair will match more than one sentence. To exploit such redundancy for more confident extraction, this paper first combines the evidence from different instances by combining their underlying components. That is, given the matched m instances $Y=\{\mathbf{y}^1, \mathbf{y}^2, \dots, \mathbf{y}^m\}$ for an entity pair (e_1, e_2) , we first decompose each instance as $\mathbf{y}^j = \mathbf{y}_1^j + \dots + \mathbf{y}_k^j + \epsilon$, then the entity-pair level decomposition $\mathbf{y} = \mathbf{y}_1 + \dots + \mathbf{y}_k + \epsilon$ is obtained by summarizing semantic components of different instances: $\mathbf{y}_i = \sum_{1 \leq j \leq m} \mathbf{y}_i^j$. Finally, the likelihood of an entity pair expressing i th relation type is scored as:

$$\text{rel}(Y, i) = \text{sim}(\mathbf{y}, \mathbf{y}_i) \text{Consistency}(\mathbf{y}, i) \log(m + 1)$$

where $\log(m + 1)$ is a score used to encourage extractions with more matching instances.

3.4 One further Issue for Distant Supervision: Training Instance Selection

The above model further provides new insights into one issue for distant supervision: *training instance selection*. In this paper, we select informative training instances by seeking a most compact subset of instances which can span the whole subspace of a relation type. That is, all instances of i th type can be represented as a linear combination of these selected instances.

However, finding the optimal subset of training instances is difficult, as there exist 2^N possible solutions for a relation type with N training instances. Therefore, this paper proposes an approximate training instance selection algorithm as follows:

- 1) Computing the centroid of i th relation type as $\mathbf{v}_i = \sum_{1 \leq j \leq n_i} \mathbf{v}_{i,j}$
- 2) Finding the set of training instances which can most compactly span the centroid by solving: $(l^1) : \mathbf{x}_i = \arg \min \|\mathbf{x}\|_1 \text{ s.t. } \|\mathbf{A}_i \mathbf{x} - \mathbf{v}_i\|_2 \leq \epsilon$
- 3) Ranking all training instances according to their absolute coefficient weight value $|\alpha_{i,j}|$;
- 4) Selecting top p percent ranked instances as final training instances.

The above training instance selection has two benefits. First, it will select informative instances and remove redundant instances: an informative instance will receive a high $|\alpha_{i,j}|$ value because many other instances can be represented using it; and if two instances are redundant, the sparse solution will only retain one of them. Second, most of the wrongly labeled training instances will be filtered, because these instances are usually not

regular expressions of i th type, so they appear only a few times and will receive a small $|\alpha_{i,j}|$.

4 Experiments

In this section, we assess the performance of our method and compare it with other methods.

Dataset. We assess our method using the KBP dataset developed by Surdeanu et al. (2012). The KBP is constructed by aligning the relations from a subset of English Wikipedia infoboxes against a document collection that merges two distinct sources: (1) a 1.5 million documents collection provided by the KBP shared task (Ji et al., 2010; Ji et al., 2011); and (2) a complete snapshot of the June 2010 version of Wikipedia. Totally 183,062 training relations and 3,334 testing relations are collected. For tuning and testing, we used the same partition as Surdeanu et al. (2012): 40 queries for development and 160 queries for formal evaluation. In this paper, each instance in KBP is represented as a feature vector using the features as the same as in (Surdeanu et al., 2012).

Baselines. We compare our method with four baselines as follows:

- **Mintz++.** This is a traditional DS assumption based model proposed by Mintz et al. (2009).
- **Hoffmann.** This is an *at-least-one* assumption based multi-instance learning method proposed by Hoffmann et al. (2011).
- **MIML.** This is a multi-instance multi-label model proposed by Surdeanu et al. (2012).
- **KNN.** This is a classical *K-Nearest-Neighbor* classifier baseline. Specifically, given an entity pair, we first classify each matching instance using the labels of its 5 (tuned on training corpus) nearest neighbors with cosine similarity, then all matching instances' classification results are added together.

Evaluation. We use the same evaluation settings as Surdeanu et al. (2012). That is, we use the official KBP scorer with two changes: (a) relation mentions are evaluated regardless of their support document; and (b) we score only on the subset of gold relations that have at least one mention in matched sentences. For evaluation, we use *Mintz++*, *Hoffmann*, and *MIML* implementation from Stanford's MIMLRE package (Surdeanu et al., 2012) and implement *KNN* by ourselves.

4.1 Experimental Results

4.1.1 Overall Results

We conduct experiments using all baselines and our semantic consistency based method. For our

method, we use top 10% weighted training instances. All features occur less than 5 times are filtered. All l^1 -minimization problems in this paper are solved using the augmented Lagrange multiplier algorithm (Yang et al., 2010), which has been proven is accurate, efficient, and robust. To select the classification threshold τ_i for i th relation type, we use the value which can achieve the best F-measure on training dataset (with an additional restriction that precision should $> 10\%$).

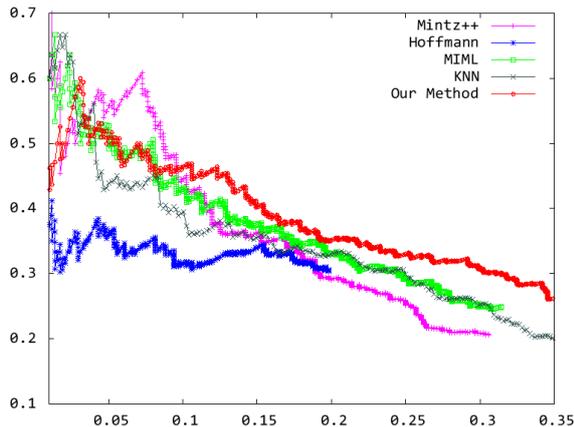


Figure 4. Precision/recall curves in KBP dataset

System	Precision	Recall	F1
Mintz++	0.260	0.250	0.255
Hoffmann	0.306	0.198	0.241
MIML	0.249	0.314	0.278
KNN	0.261	0.295	0.277
Our method	0.286	0.342	0.311

Table 1. The best F1-measures in KBP dataset

Figure 4 shows the precision/recall curves of different systems, and Table 1 shows their best F1-measures. From these results, we can see that:

1) The semantic consistency based method can achieve robust and competitive performance: in KBP dataset, our method correspondingly achieves 5.6%, 7%, 3.3% and 3.4% F1 improvements over the *Mintz++*, *Hoffmann*, *MIML* and *KNN* baselines. We believe this verifies that the semantic consistency around an instance is an effective way to identify reliable instances.

2) From Figure 4 we can see that our method achieves a consistent improvement on the high-recall region of the KBP curves (when recall > 0.1). We believe this is because by modeling the semantic consistency using the local subspace around each testing instance, our method can better solve the classification of long tail instances which are not expressed using salient patterns.

3) The local subspace around an instance can be effectively modeled as a linear subspace

spanned by training instances. From Table 1 we can see that both our method and *KNN* baseline (where the local subspace is spanned using its k nearest neighbors) achieve competitive performance: even the simple *KNN* baseline can achieve a competitive performance (0.277 in F1). This result shows: a) the effectiveness of instance-based subspace modeling; and b) by partitioning subspace into many local subspaces, the subspace model is more adaptive and robust to model prior.

4) The sparse representation is an effective way to model the local subspace using training instances. Compared with *KNN* baseline, our method can achieve a 3.4% F1 improvement. We believe this is because: (1) the discriminative nature of sparse representation as shown in (Wright et al., 2009); and (2) the sparse representation globally seeks the combination of training instances to characterize the local subspace, on contrast *KNN* uses only its nearest neighbor in the training data, which is more easily affected by noisy training instances (e.g., false positives).

4.1.2 Training Instance Selection Results

To demonstrate the effect of training instance selection, Table 2 reports our method’s performance using different proportions of training instances.

Proportion	5%	10%	20%	100%
Best F1	0.282	0.311	0.305	0.280

Table 2. The best F1-measures using different proportions of top weighted training instances

From Table 2, we can see that: ① Our training instance selection algorithm is effective: our method can achieve performance improvement using only top weighted instances. ② The training instances are highly redundant: using only 10% weighted instances can achieve a competitive performance.

5 Conclusion and Future Work

This paper proposes a semantic consistency method, which can identify reliable instances from noisy instances for distant supervised relation extraction. For future work, we want to design a more effective instance selection algorithm and embed it into our extraction framework.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants no. 61100152 and 61272324, and the National High Technology Development 863 Program of China under Grants no. 2013AA01A603.

Reference

- Andoni, Alexandr, and Piotr Indyk . 2006. *Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions*. In: Foundations of Computer Science, 2006, pp. 459-468.
- Bunescu, Razvan, and Raymond Mooney. 2007. *Learning to extract relations from the web using minimal supervision*. In: ACL 2007, pp. 576.
- Craven, Mark, and Johan Kumlien. 1999. *Constructing biological knowledge bases by extracting information from text sources*. In : Proceedings of AAAI 1999.
- Downey, Doug, Oren Etzioni, and Stephen Soderland. 2005. *A probabilistic model of redundancy in information extraction*, In: Proceeding of IJCAI 2005.
- Gupta, Rahul, and Sunita Sarawagi. 2011. *Joint training for open-domain extraction on the web: exploiting overlap when supervision is limited*. In: Proceedings of WSDM 2011, pp. 217-226.
- Hearst, Marti A. 1992. *Automatic acquisition of hyponyms from large text corpora*. In: Proceedings of COLING 1992, pp. 539-545.
- Hoffmann, Raphael, Congle Zhang, and Daniel S. Weld. 2010. *Learning 5000 relational extractors*. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, 2010, pp. 286-295.
- Hoffmann, Raphael, 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 2011, pp. 541-550.
- Ji, Heng, Ralph Grishman, Hoa Trang Dang, Kira Grifitt, and Joe Ellis. 2010. *Overview of the TAC 2010 knowledge base population track*. In: Proceedings of the Text Analytics Conference.
- Ji, Heng, Ralph Grishman, Hoa Trang Dang, Kira Grifitt, and Joe Ellis. 2011. *Overview of the TAC 2011 knowledge base population track*. In Proceedings of the Text Analytics Conference.
- Krause, Sebastian, Hong Li, Hans Uszkoreit, and Feiyu Xu. 2012. *Large-Scale learning of relation-extraction rules with distant supervision from the web*. In: ISWC 2012, pp. 263-278.
- Mintz, Mike, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. *Distant supervision for relation extraction without labeled data*. In: Proceedings ACL-AFNLP 2009, pp. 1003-1011.
- Min, Bonan, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. *Distant Supervision for Relation Extraction with an Incomplete Knowledge Base*. In: Proceedings of NAACL-HLT 2013, pp. 777-782.
- Min, Bonan, Xiang Li, Ralph Grishman, and Ang Sun. 2012. *New york university 2012 system for kbp slot filling*. In: Proceedings of TAC 2012.
- Nguyen, Truc-Vien T., and Alessandro Moschitti. 2011. *Joint distant and direct supervision for relation extraction*. In: Proceedings of IJCNLP 2011, pp. 732-740.
- Riedel, Sebastian, Limin Yao, and Andrew McCallum. 2010. *Modeling relations and their mentions without labeled text*. In: Machine Learning and Knowledge Discovery in Databases, 2010, pp. 148-163.
- Riedel, Sebastian, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. *Relation Extraction with Matrix Factorization and Universal Schemas*. In: Proceedings of NAACL-HLT 2013, pp. 74-84.
- Roth, Benjamin, and Dietrich Klakow. 2013. *Combining Generative and Discriminative Model Scores for Distant Supervision*. In: Proceedings of ACL 2013, pp. 24-29.
- Surdeanu, Mihai, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. *Multi-instance multi-label learning for relation extraction*. In: Proceedings of EMNLP-CoNLL 2012, pp. 455-465.
- Takamatsu, Shingo, Issei Sato, and Hiroshi Nakagawa. 2012. *Reducing wrong labels in distant supervision for relation extraction*. In: ACL 2012, pp. 721-729.
- Wang, Chang, Aditya Kalyanpur, James Fan, Branimir K. Boguraev, and D. C. Gondek. 2012. *Relation extraction and scoring in DeepQA*. In: IBM Journal of Research and Development, 56(3.4), pp. 9-1.
- Wang, Chang, James Fan, Aditya Kalyanpur, and David Gondek. 2011. *Relation extraction with relation topics*. In: Proceedings of EMNLP 2011, pp. 1426-1436.
- Wright, John, Allen Y. Yang, Arvind Ganesh, Shankar S. Sastry, and Yi Ma. 2009. *Robust face recognition via sparse representation*. In: Pattern Analysis and Machine Intelligence, IEEE Transactions on, 31(2), 210-227
- Wu, Fei, and Daniel S. Weld. 2007. *Autonomously semantifying wikipedia*. In: Proceedings of CIKM 2007, pp. 41-50.
- Xu, Wei, Raphael Hoffmann Le Zhao, and Ralph Grishman. 2013. *Filling Knowledge Base Gaps for Distant Supervision of Relation Extraction*. In: Proceedings of Proceedings of 2013, pp. 665-670.
- Yang, Allen Y., Shankar S. Sastry, Arvind Ganesh, and Yi Ma. 2010. *Fast l^1 -Minimization Algorithms and An Application in Robust Face Recognition: A Review*. In: Proceedings of ICIP 2010.
- Yao, Limin, Sebastian Riedel, and Andrew McCallum. 2010. *Collective cross-document relation extraction*

without labelled data. In: Proceedings of EMNLP 2010, pp. 1013-1023.

Zhang, Congle, Raphael Hoffmann, and Daniel S. Weld. 2012. *Ontological smoothing for relation extraction with minimal supervision*. In: Proceedings of AAAI 2012, pp. 157-163.

Zhang, Xingxing, Zhang, Jianwen, Zeng, Junyu, Yan, Jun, Chen, Zheng and Sui, Zhifang. 2013. *Towards Accurate Distant Supervision for Relational Facts Extraction*. In: Proceedings of ACL 2013, pp. 810-815.

Concreteness and Subjectivity as Dimensions of Lexical Meaning

Felix Hill

Computer Laboratory
Cambridge University
felix.hill@cl.cam.ac.uk

Anna Korhonen

Computer Laboratory
Cambridge University
anna.korhonen@cl.cam.ac.uk

Abstract

We quantify the lexical subjectivity of adjectives using a corpus-based method, and show for the first time that it correlates with noun concreteness in large corpora. These cognitive dimensions together influence how word meanings combine, and we exploit this fact to achieve performance improvements on the semantic classification of adjective-noun pairs.

1 Introduction

Concreteness, the degree to which language has a perceptible physical referent, and *subjectivity*, the extent to which linguistic meaning depends on the perspective of the speaker, are well established cognitive and linguistic notions. Recent results suggest that they could also be useful knowledge for natural language processing systems that aim to extract and represent the meaning of language.

Insight into concreteness can help systems to classify adjective-noun pairs according to their semantics. In the non-literal expressions *kill the process* or *black comedy*, a verb or adjective that occurs with a concrete argument in literal phrases takes an abstract argument. Turney et al. (2011) present a supervised model that exploits this effect to correctly classify 79% of adjective-noun pairs as having literal or non-literal meaning.

Subjectivity analysis has already proved highly applicable to a range of NLP applications, including sentiment analysis, information extraction and text categorization (Pang and Lee, 2004; Riloff and Wiebe, 2003). For such applications, subjectivity is analyzed at the phrasal or document level. However, recent work has highlighted the application of subjectivity analysis to lexical semantics, for instance to the tasks of disambiguating words according to their usage or sense (Wiebe and Mihalcea, 2006; Banea et al., 2014).

The importance of concreteness to NLP systems is likely to grow with the emergence of multi-modal semantic models (Bruni et al., 2012; Roller and Schulte im Walde, 2013). Such models, which learn representations from both linguistic and perceptual input, outperform text-only models on a range of evaluations. However, while multi-modal models acquire richer representations of concrete concepts, their ability to represent abstract concepts can be weaker than text-only models (Hill et al., 2013). A principled treatment of concreteness is thus likely to be important if the multi-modal approach is to prove effective on a wider range of concepts. In a similar vein, interest in subjectivity analysis is set to grow with interest in extracting sentiment and opinion from the web and social media (Benson et al., 2011). Moreover, given that humans seem to exploit both concreteness (Paivio, 1990) and subjectivity (Canestrelli et al., 2013) clues when processing language, it is likely that the same clues should benefit computational models aiming to replicate human-level performance in this area.

In this paper, we show how concreteness and subjectivity can be applied together to produce performance improvements on two classification problems: distinguishing literal and non-literal adjective-noun pairs (Turney et al., 2011), and classifying the modification type exhibited by such pairs (Boleda et al., 2012). We describe an unsupervised corpus-based method to quantify adjective subjectivity, and show that it effectively predicts the labels of a hand-coded subjectivity lexicon. Further, we show for the first time that adjective subjectivity correlates with noun concreteness in large corpora. In addition, we analyse the effect of noun concreteness and adjective subjectivity on meaning combination, illustrating how the interaction of these dimensions enables the accurate classification of adjective-noun pairs according to their semantics. We conclude by dis-

cussing other potential applications of concreteness and subjectivity to NLP.

2 Dimensions of meaning

Concreteness A large and growing body of empirical evidence indicates clear differences between concrete concepts, such as *donut* or *hot-dog* and abstract concepts, such as *guilt* or *obesity*. Concrete words are more easily learned, remembered and processed than abstract words (Paivio, 1991), while differences in brain activity (Binder et al., 2005) and cognitive representation (Hill et al., 2013) have also been observed. In linguistic constructions, concreteness appears to influence compound and phrasal semantics (Traugott, 1985; Bowdle and Gentner, 2005; Turney et al., 2011). Together with the practical applications outlined in Section 1, these facts indicate the potential value of concreteness for models aiming to replicate human performance in language processing tasks.

While automatic methods have been proposed for the quantification of lexical concreteness, they each rely on dictionaries or similar hand-coded resources (Kwong, 2008; Turney et al., 2011). We instead extract scores from a recently released dataset of lexical concepts rated on a 1-5 scale for concreteness by 20 annotators in a crowdsourcing experiment (Brysbaert et al., 2013).¹

Subjectivity Subjectivity is the degree to which language is interpretable independently of the speaker’s perspective (Langacker, 2002). For example, the utterance *he sits across the table* is more subjective than *he sits opposite Sam* as its truth depends on the speaker’s position. Language may also be subjective because it conveys evaluations or opinions (Mihalcea et al., 2007).

Computational applications of subjectivity, including sentiment analysis and information extraction, have focused largely on phrase or document meaning.² In contrast, here we present six corpus-based features designed to quantify the *lexical* subjectivity of adjectives. The features *Comparability* and *Modifiability* are identified as predictors of subjectivity by Wiebe (2000). The remainder are motivated by corpus studies and/or observations from the theoretical literature.³

¹Available at <http://crr.ugent.be/archives/1330>.

²See e.g. (Wiebe and Riloff, 2011).

³Several of the features here were applied by Hill (2012), to the task of ordering multiple-modifier strings.

Adverbiability: Quirk et al. (1985) theorizes that subjective adjectives tend to develop derived adverbial forms, whereas more objective adjectives do not. We thus define adverbiability as the frequency of derived adverbial forms relative to the frequency of their base form, e.g.

$$\frac{\sum \textit{hotly}}{\sum \textit{hot} + \sum \textit{hotly}}$$

Comparability: Wiebe (2000) observe that *gradable* are more likely to be subjective. Following Wiebe, we note that the existence of comparative forms for an adjective are indicative of gradability. We thus define comparability as the frequency of comparative or superlative forms relative to the frequency of the base form, e.g.

$$\frac{\sum \textit{hotter} + \sum \textit{hottest}}{\sum \textit{hot} + \sum \textit{hotter} + \sum \textit{hottest}}$$

LeftTendency: Adamson (2000) proposes that more subjective adjectives typically occur furthest from the noun in multiple-modifier strings such as (*hot crossed buns*). We consequently extract the LeftTendency of our adjectives, defined as the frequency of occurrence as the leftmost of two adjectives as a proportion of the overall frequency of occurrence in multiple-modifier strings.

Modifiability: Another characteristic of gradable adjectives noted by Wiebe (2000) is that they admit degree modifiers (*very/quite delicious*). We therefore extract the relative frequency of occurrence with one of a hand-coded list of English degree modifiers.

Predicativity: Bolinger (1967) proposed that subjective adjectives occur in predicative constructions (*the cake is sweet*), rather than attributive constructions (*the German capital*) more frequently than objective adjectives. We therefore extract the relative frequency of occurrence in such constructions.

Non-nominality: Many adjectives also function as nouns (*sweet cake* vs. (*boiled sweet*). Unlike nouns, many adjectives are inherently subjective, and the number of adjectives in texts correlates with human judgements of their subjectivity (Hatzivassiloglou and Wiebe, 2000). We therefore extract the frequency with which concepts are tagged as adjectives relative to as nouns, on the

assumption that ‘pure’ adjectives are on average more subjective than nominal-style adjectives.

Concreteness meets Subjectivity Demonstrable commonalities in how different people perceive the physical world suggest that concrete language may be more objective than abstract language (Langacker, 1997). Intuitively, adjectives ascribing physical properties (*wooden shed*) are more objective than those conveying abstract traits (*suspicious man*). Indeed, in certain cases the original, apparently objective, senses of polysemous adjectives are not modifiable (*very wooden shed?*), while their more abstract sense extensions are (*very wooden personality*).

Motivated by these observations, in the following sections we test two hypotheses. (1) Subjective / objective adjectives are more likely to modify abstract / concrete nouns respectively. (2) Subjectivity and concreteness can predict aspects of how adjective and noun concepts combine.

3 Analysis

In addressing (1), we extracted the 2,000 highest-frequency nouns from the Brysbaert et al. (2013) concreteness dataset. We denote by $CONC(n)$ the mean concreteness rating for noun n . For the 24,908 adjectives that occur in some adjective-noun pair with one of these nouns in the British National Corpus (BNC) (Leech et al., 1994), we extracted subjectivity features from the Google Books Corpus (Goldberg and Orwant, 2013). Since each of the six features takes values on $[0, 1]$, we define the overall subjectivity of an adjective a with feature vector $\mathbf{s}^a = [s_1^a \dots s_6^a]$ as

$$SUBJ(a) = \sum_{i=1}^6 s_i^a.$$

To verify the quality of our subjectivity features, we measured their performance as predictors in a logistic regression classifying the 3,250 adjectives labelled as subjective or not in the Wilson et al. (2005) lexicon.⁴ The combination of all features produced an overall classification accuracy of 79%. The performance of individual features as predictors in isolation is shown in Figure 1 (top).

We first tested the relationship between concreteness and subjectivity with a correlation analysis over noun concepts. For each noun n we de-

⁴Available at <http://mpqa.cs.pitt.edu/>

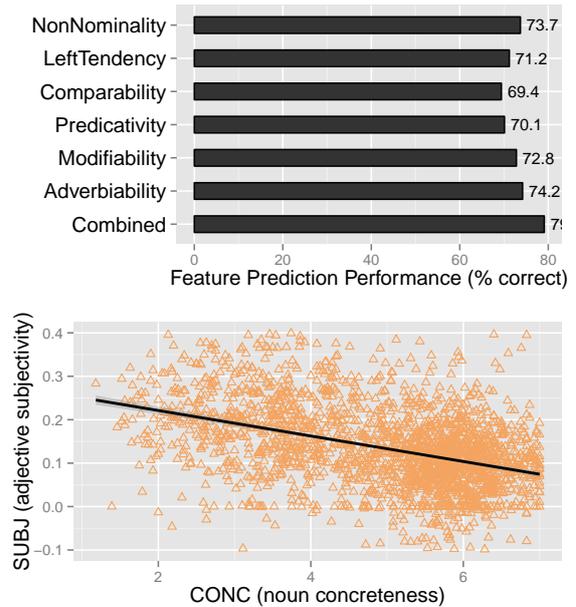


Figure 1: Top: Performance of features in predicting subjectivity labels from the Wilson et al. (2005) lexicon. Bottom: Concreteness-subjectivity correlation in adj-noun pairs.

a	$SUBJ(a)$	n	$CONC(n)$
<i>flashy</i>	1.98	<i>umbrella</i>	5
<i>honest</i>	1.63	<i>flask</i>	5
<i>good</i>	1.59	<i>automobile</i>	5
<i>Siberian</i>	6.9×10^{-4}	<i>virtue</i>	1.49
<i>interglacial</i>	6.3×10^{-4}	<i>pride</i>	1.46
<i>Soviet</i>	1.9×10^{-4}	<i>hope</i>	1.18

Table 1: The most and least subjective adjectives and most and least concrete nouns in our data.

found its *subjectivity profile* as the mean of the subjectivity vectors of its modifying adjectives

$$SUBJpfl(n) = \frac{1}{|A^n|} \sum_{a \in A^n} \mathbf{s}^a$$

where the bag A^n contains an adjective a for each occurrence of the pair (a, n) in the BNC. As hypothesized, $CONC(n)$ was a significant predictor of the magnitude of the subjectivity profile (Pearson $r = -0.421, p < 0.01$). This effect is illustrated in Figure 1 (bottom).

To explore the relationship between concreteness, subjectivity and meaning, we plotted the 20,000 highest frequency (a, n) pairs in the BNC in the $CONC$ - $SUBJ$ semantic space (Figure 2, top). In addition, to examine the effect of concreteness alone on adjective-noun semantics, we

(a, n)	Δ	(a, n)	Δ
<i>white hope</i>	4.61	<i>mature attitude</i>	4.05
<i>fresh hope</i>	4.34	<i>injured pride</i>	4.03
<i>male pride</i>	4.28	<i>black mood</i>	3.99
<i>wild hope</i>	4.06	<i>white spirit</i>	3.93

Table 2: The eight pairs with highest $\Delta = ExpCONC(a) - CONC(n)$ in our data.

defined a new adjective feature

$$ExpCONC(a) = \frac{1}{|N^a|} \sum_{n \in N^a} CONC(n)$$

where the bag N^a contains noun n for each occurrence of the pair (a, n) in the BNC. Figure 2 (bottom) illustrates the the $CONC-ExpCONC$ space.

In both spaces, the extremities reflect particular meaning combination types. Pairs in the bottom-left region of the $CONC-SUBJ$ space (objective adjectives with abstract nouns, such as *green politics*) seem to exhibit a non-literal, or at least non-prototypical modification type. In contrast, for pairs in the objective+concrete corner, the adjectives appear to perform a classifying or categorizing function (*baptist minister*).

In the $CONC-ExpCONC$ space, on the diagonal, where noun-concreteness is ‘as expected’, pairings appear to combine literally. Away from the diagonal, meaning composition is less predictable. In the top-left, where $ExpCONC$ is greater than $CONC$, the combinations are almost all non-literal, as shown in Table 2.

In this section we have outlined a set of corpus features that, taken together, enable effective approximation of adjective subjectivity. The results of our analyses also demonstrate a clear interaction between subjectivity and concreteness scores for nouns attributed by human raters. Specifically, objective adjectives are more likely to modify concrete nouns and subjective adjectives are more likely to modify abstract nouns. Qualitative investigations further suggest the interaction between these dimensions to be useful in the semantic characterization of adjective-noun pairs, a proposition we test formally in the next section.

4 Evaluation

We evaluate the potential of our adjective subjectivity features, together with noun concreteness, to predict adjective-noun semantics, based on two existing classification tasks.

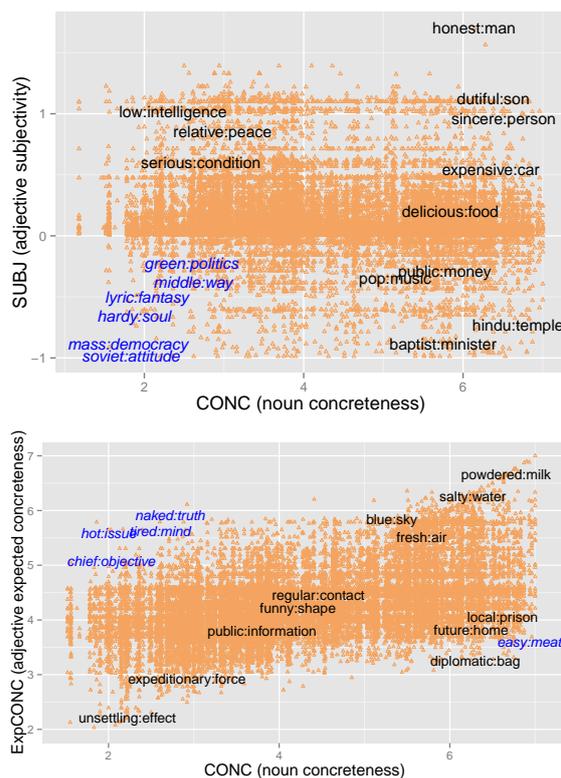


Figure 2: Adjective-noun pairs in two semantic spaces. Selected pairs are labelled for illustration, italics indicate non-literal meaning combinations.

4.1 Non-literal Composition Task

To evaluate their model of lexical concreteness, Turney et al. (2011) developed a list of 100 common adjective-noun pairs classified as either *denotative* (used literally) or *connotative* (non-literal) by five annotators. Using an identical supervised learning procedure to Turney et al. (logistic regression with 10-fold cross-validation), we test whether our lexical representations based on subjectivity and concreteness convey sufficient information to perform the same classification.

4.2 Modification-type Classification

Boleda et al. (2012) introduce a set of 370 adjective-noun pairs grouped into modification types by human judges. Because a *red car* is both a car and red, the pair is classed as *intersective*, whereas *dark humour*, which is not literally dark, is classed as *subsective*. To create a distinct but analogous binary categorization problem to the composition task, we filtered out pairs not unambiguously allocated to either class. We again aim to classify the remaining 211 intersective and 93 subsective pairs with a logistic regression.

Feature type	Composition	Modification
Baseline	55.0	69.4
Concreteness	83.0	72.7
Subjectivity	64.0	70.4
Combined	85.0	75.0
Turney et al.	79.0	-

Table 3: Prediction accuracy (%) of models with different features on the two tasks. The baseline method allocates all test pairs to the majority class.

4.3 Results

Models were trained with concreteness features (*CONC* and *ExpCONC*), subjectivity features (*SUBJ* and *SUBJpfl*) and the combination of both types (*Combined*). The performance of each model is presented in Table 3, along with a baseline score reflecting the strategy of allocating all pairs to the largest class.

On the non-literal composition task, the concreteness (83.0) and combined (85.0) models outperform that of Turney et al. (79.0). The concreteness model performance represents further confirmation of the link between concreteness and composition. The improvement of this model over Turney et al. (2011) is perhaps to be expected, since our model exploits the wide scope of the new Brysbaert et al. (2013) crowdsourced data whereas Turney et al. infer concreteness scores from a smaller training set. Notably, our combined model improves on the concreteness-only model, confirming that the interaction of concreteness and subjectivity provides additional information pertinent to meaning composition.

The modification-type task has no performance benchmark since Boleda et al. (2012) do not use their data for classification. Although all models improve on the majority-class baseline, the combined model was again most effective. Additive improvement over the baseline in each case was lower than for the composition task, which may reflect the greater subtlety inherent in the subjective/intersective classification. Indeed, inter-annotator agreement for this goldstandard (Cohen’s $\kappa = 0.87$) was lower than for the composition task (0.95), implying a less cognitively salient distinction.

5 Conclusion

We have shown that objective adjectives are most likely to modify concrete nouns, and that non-

literal combinations can emerge when this principle is violated (Section 3). Indeed, the occurrence of an adjective with a more abstract noun than those it typically modifies is a strikingly consistent indicator of metaphoricity (Table 2). In addition, we showed that both concreteness and subjectivity improve the automatic classification of adjective-noun pairs according to compositionality or modification type (Section 4). Importantly, a classifier with both subjectivity and concreteness features outperforms concreteness-only classifiers, including those proposed in previous work.

The results underline the relevance of both subjectivity and concreteness to lexical and phrasal semantics, and their application to language processing tasks. We hypothesize that concreteness and subjectivity are fundamental to human language processing because language is precisely the conveyance of information about the world from one party to another. In decoding this signal, it is clearly informative to understand to what extent the information refers directly to the world, and also to what extent it reports a fact versus an opinion. We believe these dimensions will ultimately prove essential for computational systems aiming to replicate human performance in interpreting language. As the results suggest, they may be particularly important for capturing the intricacies of semantic composition and thus extending representations beyond the lexeme.

Of course, two dimensions alone are not sufficient to reflect all of the subtleties of adjective and noun semantics. For instance, our model classifies *white spirit*, a transparent cleaning product, as non-literal, since the lexical concreteness score does not allow for strong noun polysemy. Further, it makes no allowance for wider sentential context, which can be an important clue to modification type in such cases.

We aim to address these limitations in future work by integrating subjectivity and concreteness with conventionally acquired semantic representations, and, ultimately, models that integrate input corresponding to the perceptual modalities.

6 Acknowledgements

The authors are supported by St John’s College, Cambridge and The Royal Society.

References

- Sylvia Adamson. 2000. A lovely little example. In Olga Fischer, Annette Rosenbach, and Deiter Stein, editors, *Pathways of change: Grammaticalization in English*. John Benjamins.
- Carmen Banea, Rada Mihalcea, and Janyce Wiebe. 2014. Sense-level subjectivity in a multilingual setting. *Computer Speech & Language*, 28(1):7–19.
- Edward Benson, Aria Haghighi, and Regina Barzilay. 2011. Event discovery in social media feeds. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 389–398. Association for Computational Linguistics.
- Jeffrey R Binder, Chris F Westbury, Kristen A McKiernan, Edward T Possing, and David A Medler. 2005. Distinct brain systems for processing concrete and abstract concepts. *Journal of Cognitive Neuroscience*, 17(6):905–917.
- Gemma Boleda, Eva Maria Vecchi, Miquel Cornudella, and Louise McNally. 2012. First-order vs. higher-order modification in distributional semantics. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1223–1233. Association for Computational Linguistics.
- Dwight Bolinger. 1967. Adjectives in english: attribution and predication. *Lingua*, 18:1–34.
- Brian F Bowdle and Dedre Gentner. 2005. The career of metaphor. *Psychological review*, 112(1):193.
- Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 136–145. Association for Computational Linguistics.
- Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2013. Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior research methods*, pages 1–8.
- Anneloes R Canestrelli, Willem M Mak, and Ted JM Sanders. 2013. Causal connectives in discourse processing: How differences in subjectivity are reflected in eye movements. *Language and Cognitive Processes*, 28(9):1394–1413.
- Yoav Goldberg and Jon Orwant. 2013. A dataset of syntactic-ngrams over time from a very large corpus of english books. In *Second Joint Conference on Lexical and Computational Semantics, Association for Computational Linguistics*, pages 241–247. Association for Computational Linguistics.
- Vasileios Hatzivassiloglou and Janyce M Wiebe. 2000. Effects of adjective orientation and gradability on sentence subjectivity. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pages 299–305. Association for Computational Linguistics.
- Felix Hill, Douwe Kiela, and Anna Korhonen. 2013. Concreteness and corpora: A theoretical and practical analysis. *ACL 2013 Workshop on Cognitive Modelling and Computational Linguistics, CMCL 2013*, page 75.
- Felix Hill. 2012. Beauty before age? Applying subjectivity to automatic english adjective ordering. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Student Research Workshop*, pages 11–16. Association for Computational Linguistics.
- Oi Yee Kwong. 2008. A preliminary study on the impact of lexical concreteness on word sense disambiguation. In *PACLIC*, pages 235–244.
- Ronald W Langacker. 1997. Consciousness, construal and subjectivity. *Language structure, discourse and the access to consciousness. Advances in Consciousness Research*. John Benjamins, pages 49–57.
- Ronald W Langacker. 2002. Deixis and subjectivity. In Frank Brisard, editor, *Grounding: The epistemic footing of deixis and reference*, pages 1–28. De Gruyter.
- Geoffrey Leech, Roger Garside, and Michael Bryant. 1994. Claws4: the tagging of the british national corpus. In *Proceedings of ACL*, pages 622–628. Association for Computational Linguistics.
- Rada Mihalcea, Carmen Banea, and Janyce Wiebe. 2007. Learning multilingual subjective language via cross-lingual projections. In *Annual Meeting of the Association for Computational Linguistics*, volume 45, page 976.
- Allan Paivio. 1990. *Mental Representations: A Dual Coding Approach*. Oxford University Press.
- Allan Paivio. 1991. Dual coding theory: Retrospect and current status. *Canadian Journal of Psychology/Revue canadienne de psychologie*, 45(3):255.
- Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics.
- Randolph Quirk, David Crystal, and Pearson Education. 1985. *A Comprehensive Grammar of the English Language*, volume 397. Cambridge Univ Press.
- Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 conference on Empirical methods in*

- natural language processing*, pages 105–112. Association for Computational Linguistics.
- Stephen Roller and Sabine Schulte im Walde. 2013. A multimodal LDA model integrating textual, cognitive and visual modalities. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1146–1157, Seattle, Washington, USA. Association for Computational Linguistics.
- Elizabeth C Traugott. 1985. On regularity in semantic change. *Journal of literary semantics*, 14(3):155–173.
- Peter D Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the 2011 Conference on the Empirical Methods in Natural Language Processing*, pages 680–690.
- Janyce Wiebe and Rada Mihalcea. 2006. Word sense and subjectivity. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1065–1072. Association for Computational Linguistics.
- Janyce Wiebe and Ellen Riloff. 2011. Finding mutual benefit between subjectivity analysis and information extraction. *Affective Computing, IEEE Transactions on*, 2(4):175–191.
- Janyce Wiebe. 2000. Learning subjective adjectives from corpora. In *AAAI/IAAI*, pages 735–740.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 347–354. Association for Computational Linguistics.

Infusion of Labeled Data into Distant Supervision for Relation Extraction

Maria Pershina⁺ Bonan Min^{^*} Wei Xu[#] Ralph Grishman⁺

⁺New York University, New York, NY

{pershina, grishman}@cs.nyu.edu

[^]Raytheon BBN Technologies, Cambridge, MA

bmin@bbn.com

[#]University of Pennsylvania, Philadelphia, PA

xwe@cis.upenn.edu

Abstract

Distant supervision usually utilizes only unlabeled data and existing knowledge bases to learn relation extraction models. However, in some cases a small amount of human labeled data is available. In this paper, we demonstrate how a state-of-the-art multi-instance multi-label model can be modified to make use of these reliable sentence-level labels in addition to the relation-level distant supervision from a database. Experiments show that our approach achieves a statistically significant increase of 13.5% in F-score and 37% in area under the precision recall curve.

1 Introduction

Relation extraction is the task of tagging semantic relations between pairs of entities from free text. Recently, distant supervision has emerged as an important technique for relation extraction and has attracted increasing attention because of its effective use of readily available databases (Mintz et al., 2009; Bunescu and Mooney, 2007; Snyder and Barzilay, 2007; Wu and Weld, 2007). It automatically labels its own training data by heuristically aligning a knowledge base of facts with an unlabeled corpus. The intuition is that any sentence which mentions a pair of entities (e_1 and e_2) that participate in a relation, r , is likely to express the fact $r(e_1, e_2)$ and thus forms a positive training example of r .

One of most crucial problems in distant supervision is the inherent errors in the automatically generated training data (Roth et al., 2013). Table 1 illustrates this problem with a toy example. Sophisticated multi-instance learning algorithms (Riedel et al., 2010; Hoffmann et al., 2011;

* Most of the work was done when this author was at New York University

Surdeanu et al., 2012) have been proposed to address the issue by loosening the distant supervision assumption. These approaches consider all mentions of the same pair (e_1, e_2) and assume that *at-least-one* mention actually expresses the relation. On top of that, researchers further improved performance by explicitly adding preprocessing steps (Takamatsu et al., 2012; Xu et al., 2013) or additional layers inside the model (Ritter et al., 2013; Min et al., 2013) to reduce the effect of training noise.

True Positive	... to get information out of captured al-Qaida leader Abu Zubaydah .
False Positive	... Abu Zubaydah and former Taliban leader Jalaluddin Haqqani ...
False Negative	... Abu Zubaydah is one of Osama bin Laden's <i>senior operational planners</i> ...

Table 1: Classic errors in the training data generated by a toy knowledge base of only one entry personTitle(Abu Zubaydah, *leader*).

However, the potential of these previously proposed approaches is limited by the inevitable gap between the relation-level knowledge and the instance-level extraction task. In this paper, we present the first effective approach, Guided DS (distant supervision), to incorporate labeled data into distant supervision for extracting relations from sentences. In contrast to simply taking the union of the hand-labeled data and the corpus labeled by distant supervision as in the previous work by Zhang et al. (2012), we generalize the labeled data through feature selection and model this additional information directly in the latent variable approaches. Aside from previous semi-supervised work that employs labeled and unlabeled data (Yarowsky, 2013; Blum and Mitchell, 1998; Collins and Singer, 2011; Nigam, 2001, and others), this is a learning scheme that combines unlabeled text and two training sources whose quantity and quality are radically different (Liang et al., 2009).

To demonstrate the effectiveness of our pro-

Guideline $g = \{g_i i = 1, 2, 3\}$: types of entities, <i>dependency path</i> , span word (optional)	Relation $r(g)$
person_person, $nsubj \rightarrow \leftarrow dobj$, married	personSpouse
person_organization, $nsubj \rightarrow \leftarrow prep_of$, became	personMemberOf
organization_organization, $nsubj \rightarrow \leftarrow prep_of$, company	organizationSubsidiaries
person_person, $poss \rightarrow \leftarrow appos$, sister	personSiblings
person_person, $poss \rightarrow \leftarrow appos$, father	personParents
person_title, $\leftarrow nn$	personTitle
organization_person, $prep_of \rightarrow appos \rightarrow$	organizationTopMembersEmployees
person_cause, $nsubj \rightarrow \leftarrow prep_of$	personCauseOfDeath
person_number, $\leftarrow appos$	personAge
person_date, $nsubjpass \rightarrow \leftarrow prep_on \leftarrow num$	personDateOfBirth

Table 2: Some examples from the final set \mathbf{G} of extracted guidelines.

posed approach, we extend MIML (Surdeanu et al., 2012), a state-of-the-art distant supervision model and show a significant improvement of 13.5% in F-score on the relation extraction benchmark TAC-KBP (Ji and Grishman, 2011) dataset. While prior work employed tens of thousands of human labeled examples (Zhang et al., 2012) and only got a 6.5% increase in F-score over a logistic regression baseline, our approach uses much less labeled data (about 1/8) but achieves much higher improvement on performance over stronger baselines.

2 The Challenge

Simply taking the union of the hand-labeled data and the corpus labeled by distant supervision is not effective since hand-labeled data will be swamped by a larger amount of distantly labeled data. An effective approach must recognize that the hand-labeled data is more reliable than the automatically labeled data and so must take precedence in cases of conflict. Conflicts cannot be limited to those cases where all the features in two examples are the same; this would almost never occur, because of the dozens of features used by a typical relation extractor (Zhou et al., 2005). Instead we propose to perform feature selection to generalize human labeled data into *training guidelines*, and integrate them into latent variable model.

2.1 Guidelines

The sparse nature of feature space dilutes the discriminative capability of useful features. Given the small amount of hand-labeled data, it is important to identify a small set of features that are general enough while being capable of predicting quite accurately the type of relation that may hold between two entities.

We experimentally tested alternative feature sets by building supervised Maximum Entropy (MaxEnt) models using the hand-labeled data (Table 3), and selected an effective combination of three features from the full feature set used by Surdeanu et al., (2011):

- the semantic types of the two arguments (e.g. person, organization, location, date, title, ...)
- the sequence of dependency relations along the path connecting the heads of the two arguments in the dependency tree.
- a word in the sentence between the two arguments

These three features are strong indicators of the type of relation between two entities. In some cases the semantic types of the arguments alone narrows the possibilities to one or two relation types. For example, entity types such as person and title often implies the relation personTitle. Some lexical items are clear indicators of particular relations, such as “brother” and “sister” for a sibling relationship

We extract guidelines from hand-labeled data. Each guideline $g = \{g_i | i = 1, 2, 3\}$ consists of a pair of semantic types, a dependency path, and optionally a span word and is associated with a particular relation $r(g)$. We keep only those guidelines

Model	Precision	Recall	F-score
MaxEnt ^{all}	18.6	6.3	9.4
MaxEnt ^{two}	24.13	10.75	14.87
MaxEnt ^{three}	40.27	12.40	18.97

Table 3: Performance of a MaxEnt, trained on hand-labeled data using all features (Surdeanu et al., 2011) vs using a subset of two (types of entities, dependency path), or three (adding a span word) features, and evaluated on the test set.

which make the correct prediction for *all* and at least $k=3$ examples in the training corpus (threshold 3 was obtained by running experiments on the development dataset). Table 2 shows some examples in the final set \mathbf{G} of extracted guidelines.

3 Guided DS

Our goal is to jointly model human-labeled ground truth and structured data from a knowledge base in distant supervision. To do this, we extend the MIML model (Surdeanu et al., 2012) by adding a new layer as shown in Figure 1.

The input to the model consists of (1) distantly supervised data, represented as a list of n bags¹ with a vector \mathbf{y}_i of binary gold-standard labels, either *Positive*(P) or *Negative*(N) for each relation $r \in R$; (2) generalized human-labeled ground truth, represented as a set \mathbf{G} of feature conjunctions $g = \{g_i | i=1,2,3\}$ associated with a unique relation $r(g)$. Given a bag of sentences, \mathbf{x}_i , which mention an i th entity pair (e_1, e_2), our goal is to correctly predict which relation is mentioned in each sentence, or *NR* if none of the relations under consideration are mentioned. The vector \mathbf{z}_i contains the latent mention-level classifications for the i th entity pair. We introduce a set of latent variables \mathbf{h}_i which model human ground truth for each mention in the i th bag and take precedence over the current model assignment \mathbf{z}_i .

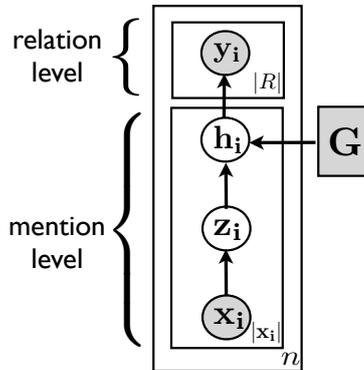


Figure 1: Plate diagram of Guided DS

Let i, j be the index in the bag and the mention level, respectively. We model mention-level extraction $p(z_{ij} | \mathbf{x}_{ij}; \mathbf{w}_z)$, human relabeling $h_{ij}(\mathbf{x}_{ij}, z_{ij})$ and multi-label aggregation $p(y_i^r | \mathbf{h}_i; \mathbf{w}_y)$. We define:

- $y_i^r \in \{P, N\}$: r holds for the i th bag or not.
- \mathbf{x}_{ij} is the feature representation of the j th relation mention in the i th bag. We use the same set of features as in Surdeanu et al. (2012).

¹A bag is a set of mentions sharing same entity pair.

- $z_{ij} \in R \cup NR$: a latent variable that denotes the relation of the j th mention in the i th bag
- $h_{ij} \in R \cup NR$: a latent variable that denotes the refined relation of the mention \mathbf{x}_{ij}

We define relabeled relations h_{ij} as following:

$$h_{ij}(\mathbf{x}_{ij}, z_{ij}) = \begin{cases} r(g), & \text{if } \exists! g \in \mathbf{G} \text{ s.t. } g = \{g_k\} \subseteq \{\mathbf{x}_{ij}\} \\ z_{ij}, & \text{otherwise} \end{cases}$$

Thus, relation $r(g)$ is assigned to h_{ij} iff there exists a unique guideline $g \in \mathbf{G}$, such that the feature vector \mathbf{x}_{ij} contains all constituents of g , i.e. entity types, a dependency path and maybe a span word, if g has one. We use mention relation z_{ij} inferred by the model only in case no such a guideline exists or there is more than one matching guideline. We also define:

- \mathbf{w}_z is the weight vector for the multi-class relation mention-level classifier²
- \mathbf{w}_y^r is the weight vector for the r th binary top-level aggregation classifier (from mention labels to bag-level prediction). We use \mathbf{w}_y to represent $\mathbf{w}_y^1, \mathbf{w}_y^2, \dots, \mathbf{w}_y^{|R|}$.

Our approach is aimed at improving the mention-level classifier, while keeping the multi-instance multi-label framework to allow for joint modeling.

4 Training

We use a hard expectation maximization algorithm to train the model. Our objective function is to maximize log-likelihood of the data:

$$\begin{aligned} LL(\mathbf{w}_y, \mathbf{w}_z) &= \sum_{i=1}^n \log p(\mathbf{y}_i | \mathbf{x}_i, \mathbf{w}_y, \mathbf{w}_z, \mathbf{G}) \\ &= \sum_{i=1}^n \log \sum_{\mathbf{h}_i} p(\mathbf{y}_i, \mathbf{h}_i | \mathbf{x}_i, \mathbf{w}_y, \mathbf{w}_z, \mathbf{G}) \\ &= \sum_{i=1}^n \log \sum_{\mathbf{h}_i} \prod_{j=1}^{|\mathbf{h}_i|} p(h_{ij} | \mathbf{x}_{ij}, \mathbf{w}_z, \mathbf{G}) \prod_{r \in P_i \cup N_i} p(y_i^r | \mathbf{h}_i, \mathbf{w}_y^r) \end{aligned}$$

where the last equality is due to conditional independence. Because of the non-convexity of $LL(\mathbf{w}_y, \mathbf{w}_z)$ we approximate and maximize the joint log-probability $p(\mathbf{y}_i, \mathbf{h}_i | \mathbf{x}_i, \mathbf{w}_y, \mathbf{w}_z, \mathbf{G})$ for each entity pair in the database:

$$\begin{aligned} &\log p(\mathbf{y}_i, \mathbf{h}_i | \mathbf{x}_i, \mathbf{w}_y, \mathbf{w}_z, \mathbf{G}) \\ &= \sum_{j=1}^{|\mathbf{h}_i|} \log p(h_{ij} | \mathbf{x}_{ij}, \mathbf{w}_z, \mathbf{G}) + \sum_{r \in P_i \cup N_i} \log p(y_i^r | \mathbf{h}_i, \mathbf{w}_y^r). \end{aligned}$$

²All classifiers are implemented using L2-regularized logistic regression with Stanford CoreNLP package.

Iteration	1	2	3	4	5	6	7	8
(a) Corrected relations:	2052	718	648	596	505	545	557	535
(b) Retrieved relations:	10219	860	676	670	621	599	594	592
Total relabelings	12271	1578	1324	1264	1226	1144	1153	1127

Table 4: Number of relabelings for each training iteration of Guided DS: (a) relabelings due to corrected relations, e.g. `personChildren` \rightarrow `personSiblings` (b) relabelings due to retrieved relations, e.g. `notRelated(NR)` \rightarrow `personTitle`

Algorithm 1 : Guided DS training

```

1: Phase 1: build set  $\mathbf{G}$  of guidelines
2: Phase 2: EM training
3: for iteration = 1, ...,  $T$  do
4:   for  $i = 1, \dots, n$  do
5:     for  $j = 1, \dots, |\mathbf{x}_i|$  do
6:        $z_{ij}^* = \operatorname{argmax}_{z_{ij}} p(z_{ij} | \mathbf{x}_i, \mathbf{y}_i, \mathbf{w}_z, \mathbf{w}_y)$ 
7:        $h_{ij}^* = \begin{cases} r(g), & \text{if } \exists! g \in \mathbf{G} : \{g_k\} \subseteq \{\mathbf{x}_{ij}\} \\ z_{ij}^*, & \text{otherwise} \end{cases}$ 
8:       update  $\mathbf{h}_i$  with  $h_{ij}^*$ 
9:     end for
10:   end for
11:    $\mathbf{w}_z^* = \operatorname{argmax}_{\mathbf{w}_z} \sum_{i=1}^n \sum_{j=1}^{|\mathbf{x}_i|} \log p(h_{ij} | \mathbf{x}_{ij}, \mathbf{w}_z)$ 
12:   for  $r \in R$  do
13:      $\mathbf{w}_y^{r*} = \operatorname{argmax}_{\mathbf{w}_y} \sum_{1 \leq i \leq n \text{ s.t. } r \in P_i \cup N_i} \log p(y_i^r | \mathbf{h}_i, \mathbf{w}_y)$ 
14:   end for
15: end for
16: return  $\mathbf{w}_z, \mathbf{w}_y$ 

```

The pseudocode is presented as algorithm 1.

The following approximation is used for inference at step 6:

$$\begin{aligned}
p(z_{ij} | \mathbf{x}_i, \mathbf{y}_i, \mathbf{w}_z, \mathbf{w}_y) &\propto p(\mathbf{y}_i, z_{ij} | \mathbf{x}_i, \mathbf{w}_z, \mathbf{w}_y) \\
&\approx p(z_{ij} | x_{ij}, \mathbf{w}_z) p(\mathbf{y}_i | \mathbf{h}'_i, \mathbf{w}_y) \\
&= p(z_{ij} | x_{ij}, \mathbf{w}_z) \prod_{r \in P_i \cup N_i} p(y_i^r | \mathbf{h}'_i, \mathbf{w}_y^r),
\end{aligned}$$

where \mathbf{h}'_i contains previously inferred and maybe further relabeled mention labels for group i (steps 5-10), with the exception of component j whose label is replaced by z_{ij} . In the M-step (lines 12-15) we optimize model parameters $\mathbf{w}_z, \mathbf{w}_y$, given the current assignment of mention-level labels \mathbf{h}_i .

Experiments show that Guided DS efficiently learns new model, resulting in a drastically decreasing number of needed relabelings for further iterations (Table 4). At the inference step we first classify all mentions:

$$z_{ij}^* = \operatorname{argmax}_{z \in R \cup NR} p(z | x_{ij}, \mathbf{w}_z)$$

Then final relation labels for i th entity tuple are

obtained via the top-level classifiers:

$$y_i^{r*} = \operatorname{argmax}_{y \in \{P, N\}} p(y | \mathbf{z}_i^*, \mathbf{w}_y^r)$$

5 Experiments

5.1 Data

We use the KBP (Ji and Grishman, 2011) dataset³ which is preprocessed by Surdeanu et al. (2011) using the Stanford parser⁴ (Klein and Manning, 2003). This dataset is generated by mapping Wikipedia infoboxes into a large unlabeled corpus that consists of 1.5M documents from KBP source corpus and a complete snapshot of Wikipedia.

The KBP 2010 and 2011 data includes 200 query named entities with the relations they are involved in. We used 40 queries as development set and the rest 160 queries (3334 entity pairs that express a relation) as the test set. The official KBP evaluation is performed by pooling the system responses and manually reviewing each response, producing a hand-checked assessment data. We used KBP 2012 assessment data to generate guidelines since queries from different years do not overlap. It contains about 2500 labeled sentences of 41 relations, which is less than 0.09% of the size of the distantly labeled dataset of 2M sentences. The final set \mathbf{G} consists of 99 guidelines (section 2.1).

5.2 Models

We implement Guided DS on top of the MIML (Surdeanu et al., 2012) code base⁵. Training MIML on a simple fusion of distantly-labeled and human-labeled datasets does not improve the maximum F-score since this hand-labeled data is swamped by a much larger amount of distant-supervised data of much lower quality. Upsampling the labeled data did not improve the performance either. We experimented with different up-sampling ratios and report best results using ratio 1:1 in Figure 2.

³Available from Linguistic Data Consortium (LDC) at <http://projects.ldc.upenn.edu/kbp/data>.

⁴<http://nlp.stanford.edu/software/lex-parser.shtml>

⁵Available at <http://nlp.stanford.edu/software/mimlre.shtml>.

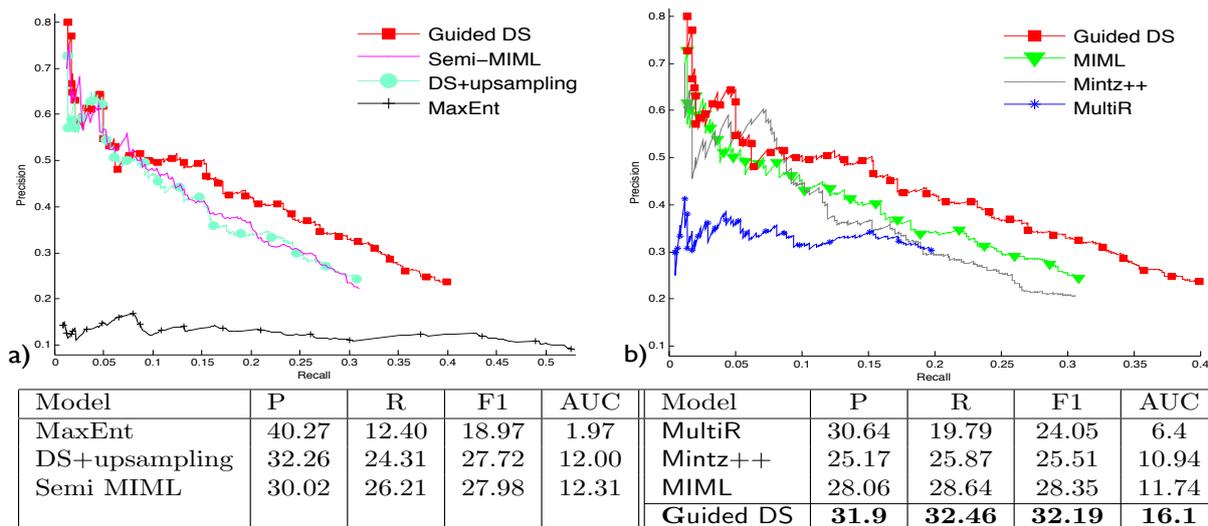


Figure 2: Performance of **Guided DS** on KBP task compared to a) baselines: MaxEnt, DS+upsampling, Semi-MIML (Min et al., 2013) b) state-of-art models: Mintz++ (Mintz et al., 2009), MultiR (Hoffmann et al., 2011), MIML (Surdeanu et al., 2012)

Our baselines: 1) MaxEnt is a supervised maximum entropy baseline trained on a human-labeled data; 2) DS+upsampling is an upsampling experiment, where MIML was trained on a mix of a distantly-labeled and human-labeled data; 3) Semi-MIML is a recent semi-supervised extension. We also compare **Guided DS** with three state-of-the-art models: 1) MultiR and 2) MIML are two distant supervision models that support multi-instance learning and overlapping relations; 3) Mintz++ is a single-instance learning algorithm for distant supervision. The difference between **Guided DS** and all other systems is significant with p -value less than 0.05 according to a paired t -test assuming a normal distribution.

5.3 Results

We scored our model against all 41 relations and thus replicated the actual KBP evaluation. Figure 2 shows that our model consistently outperforms all six algorithms at almost all recall levels and improves the maximum F -score by more than 13.5% relative to MIML (from 28.35% to 32.19%) as well as increases the area under precision-recall curve by more than 37% (from 11.74 to 16.1). Also, **Guided DS** improves the overall recall by more than 9% absolute (from 30.9% to 39.93%) at a comparable level of precision (24.35% for MIML vs 23.64% for **Guided DS**), while increases the running time of MIML by only 3%. Thus, our approach outperforms state-of-the-art model for relation extraction using much less labeled data that was used by Zhang et al., (2012) to outper-

form logistic regression baseline. Performance of **Guided DS** also compares favorably with best scored hand-coded systems for a similar task such as Sun et al., (2011) system for KBP 2011, which reports an F -score of 25.7%.

6 Conclusions and Future Work

We show that relation extractors trained with distant supervision can benefit significantly from a small number of human labeled examples. We propose a strategy to generate and select guidelines so that they are more generalized forms of labeled instances. We show how to incorporate these guidelines into an existing state-of-art model for relation extraction. Our approach significantly improves performance in practice and thus opens up many opportunities for further research in RE where only a very limited amount of labeled training data is available.

Acknowledgments

Supported by the Intelligence Advanced Research Projects Activity (IARPA) via Air Force Research Laboratory (AFRL) contract number FA8650-10-C-7058. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, AFRL, or the U.S. Government.

References

- Avrim Blum and Tom M. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT)*, pages 92–100.
- Razvan C. Bunescu and Raymond J. Mooney. 2007. Learning to extract relations from the web using minimal supervision. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*.
- Michael Collins and Yorav Singer. 1999. Unsupervised models for named entity classification. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-VLC)*.
- Mark Craven and Johan Kumlien. 1999. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology (ISMB)*, pages 77–86.
- Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. 2008. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74.
- Raphael Hoffmann, Congle Zhang, and Daniel S. Weld. 2010. Learning 5000 relational extractors. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 286–295.
- Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke S. Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 541–550.
- Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1148–1158.
- Heng Ji, Ralph Grishman, and Hoa Trang Dang. 2011. Overview of the TAC-2011 knowledge base population track. In *Text Analysis Conference Workshop*.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41th Annual Meeting of the Association for Computational Linguistics (ACL)*.
- Percy Liang, Michael I. Jordan and Dan Klein. 2009. Learning From Measurements in Exponential Families. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pages = 641–648
- 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 the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.
- Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing (ACL)*, pages 1003–1011.
- Ramesh Nallapati. 2004. Discriminative models for information retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 64–71.
- Truc-Vien T. Nguyen and Alessandro Moschitti. 2011. End-to-end relation extraction using distant supervision from external semantic repositories. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 277–282.
- Kamal Paul Nigam. 2001. Using Unlabeled Data to Improve Text Classification. Ph.D. thesis, School of Computer Science, Carnegie Mellon University.
- Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*, pages 148–163.
- Alan Ritter, Luke Zettlemoyer, Mausam, and Oren Etzioni. 2013. Modeling missing data in distant supervision for information extraction. *Transactions of the Association for Computational Linguistics*.
- Benjamin Roth, Tassilo Barth, Michael Wiegand, and Dietrich Klakow. 2013. A Survey of Noise Reduction Methods for Distant Supervision. In *Proceedings of Conference on Information and Knowledge Management (CIKM-AKBC)*.
- Benjamin Snyder and Regina Barzilay. 2007. Database-text alignment via structured multilabel classification. In *Proceedings of IJCAI*.
- Ang Sun, Ralph Grishman, Wei Xu, and Bonan Min. 2011. New york university 2011 system for kbp slot filling. In *Text Analysis Conference (TAC-KBP)*.
- Mihai Surdeanu, J. Turmo, and A. Ageo. 2006. A hybrid approach for the acquisition of information extraction patterns. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics Workshop on Adaptive Text Extraction and Mining (EACL)*.
- Mihai Surdeanu, Sonal Gupta, John Bauer, David McClosky, Angel X. Chang, Valentin I. Spitskovsky, and Christopher D. Manning. 2011. Stanford’s

- Distantly-Supervised Slot-Filling System. In *Proceedings of the Text Analysis Conference (TAC-KBP)*.
- Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 455–465.
- Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 721–729.
- Fei Wu and Daniel S. Weld. 2007. Autonomously semantifying wikipedia. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, pages 41–50.
- Wei Xu, Raphael Hoffmann, Zhao Le, and Ralph Grishman. 2013. Filling knowledge base gaps for distant supervision of relation extraction. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics (ACL)*.
- David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33th Annual Meeting of the Association for Computational Linguistics (ACL)*.
- Ce Zhang, Feng Niu, Christopher Ré, and Jude Shavlik. 2012. Big data versus the crowd: Looking for relationships in all the right places. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 825–834. Association for Computational Linguistics.
- Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Recognizing Implied Predicate-Argument Relationships in Textual Inference

Asher Stern

Computer Science Department
Bar-Ilan University
astern7@cs.biu.ac.il

Ido Dagan

Computer Science Department
Bar-Ilan University
dagan@cs.biu.ac.il

Abstract

We investigate recognizing implied predicate-argument relationships which are not explicitly expressed in syntactic structure. While prior works addressed such relationships as an extension to semantic role labeling, our work investigates them in the context of textual inference scenarios. Such scenarios provide prior information, which substantially eases the task. We provide a large and freely available evaluation dataset for our task setting, and propose methods to cope with it, while obtaining promising results in empirical evaluations.

1 Motivation and Task

This paper addresses a typical sub-task in textual inference scenarios, of recognizing implied predicate-argument relationships which are not expressed explicitly through syntactic structure. Consider the following example:

- The crucial role Vioxx plays in Merck's portfolio was apparent last week when Merck's shares plunged 27 percent to 33 dollars after the withdrawal announcement.
- (i)

While a human reader understands that the withdrawal refers to Vioxx, and hence an implied predicate-argument relationship holds between them, this relationship is not expressed in the syntactic structure, and will be missed by syntactic parsers or standard semantic role labelers.

This paper targets such types of implied relationships in textual inference scenarios. Particularly, we investigate the setting of *Recognizing Textual Entailment (RTE)* as a typical scenario of textual inference. We suggest, however, that the same challenge, as well as the solutions proposed in our work, are applicable, with proper adaptations, to other textual-inference scenarios, like

Question Answering, and *Information Extraction* (see Section 6).

An RTE problem instance is composed of two text fragments, termed *Text* and *Hypothesis*, as input. The task is to recognize whether a human reading the Text would infer that the Hypothesis is most likely true (Dagan et al., 2006). For our problem, consider a positive Text Hypothesis pair, where the Text is example (i) above and the Hypothesis is:

- (ii) Merck withdrew Vioxx.

A common approach for recognizing textual entailment is to verify that all the textual elements of the Hypothesis are *covered*, or *aligned*, by elements of the Text. These elements typically include lexical terms as well as relationships between them. In our example, the Hypothesis lexical terms (“Merck”, “withdrew” and “Vioxx”) are indeed covered by the Text. Yet, the predicate-argument relationships (e.g., “withdrawal-Vioxx”) are not expressed in the text explicitly. In such a case, an RTE system has to verify that the predicate-argument relationships which are explicitly expressed in the Hypothesis, are *implied* from the Text discourse. Such cases are quite frequent (~17%) in the settings of our dataset, described in Section 3.

Consequently, we define the task of recognizing implied predicate-argument relationships, with illustrating examples in Table 1, as follows. The input includes a *Text* and a *Hypothesis*. Two terms in the Hypothesis, *predicate* and *argument*, are marked, where a predicate-argument relationship between them is explicit in the Hypothesis syntactic structure. Two terms in the Text, *candidate-predicate* and *candidate-argument*, aligned to the Hypothesis predicate and argument, are marked as well. However, no predicate-argument relationship between them is expressed syntactically. The task is to recognize whether the predicate-

#	Hypothesis	Text	Y/N
1	Merck [withdrew] _{pred} [Vioxx] _{arg} from the market.	The crucial role [Vioxx] _{cand-arg} plays in Merck’s portfolio was apparent last week when Merck’s shares plunged 27 percent to 33 dollars after the [withdrawal] _{cand-pred} announcement.	Y
2	Barbara Cummings heard the tale of a woman who was coming to Crawford to [join] _{pred} Cindy Sheehans [protest] _{arg} .	Sheehan’s [protest] _{cand-arg} is misguided and is hurting troop morale. . . . Sheehan never wanted Casey to [join] _{cand-pred} the military.	N
3	Casey Sheehan was [killed] _{pred} in [Iraq] _{arg} .	5 days after he arrived in [Iraq] _{cand-arg} last year, Casey Sheehan was [killed] _{cand-pred} .	Y
4	Hurricane Rita [threatened] _{pred} [New Orleans] _{arg} .	Hurricane Rita was upgraded from a tropical storm as it [threatened] _{cand-pred} the <u>southeastern United States</u> , forcing an alert in southern Florida and scuttling plans to repopulate [New Orleans] _{cand-arg} after Hurricane Katrina turned it into a ghost city 3 weeks earlier.	Y
5	Alberto Gonzales defends [renewal] _{pred} of the [Patriot Act] _{arg} to Congress.	A senior official defended the [Patriot Act] _{cand-arg} President Bush has urged Congress to [renew] _{cand-pred} <u>the law</u>	Y
6	The [train] _{arg} [crash] _{pred} injured nearly 200 people.	At least 10 people were killed . . . in the [crash] _{cand-pred} Alvarez is accused of . . . causing the derailment of one [train] _{cand-arg}	Y

Table 1: Example task instances from our dataset. The last column specifies the Yes/No annotation, indicating whether the sought predicate-argument relationship is implied in the Text. For illustration, a dashed line indicates an explicit argument that is related to the candidate argument through some kind of discourse reference. Pred, arg and cand abbreviate predicate, argument and candidate respectively.

argument relationship, as expressed in the Hypothesis, holds implicitly also in the Text.

To address this task, we provide a large and freely available annotated dataset, and propose methods for coping with it. A related task, described in the next section, deals with such implied predicate-argument relationships as an extension to *Semantic Role Labeling*. While the results reported so far on that annotation task were relatively low, we suggest that the task itself may be more complicated than what is actually required in textual inference scenarios. On the other hand, the results obtained for our task, which does fit textual inference scenarios, are promising, and encourage utilizing algorithms for this task in actual inference systems.

2 Prior Work

The most notable work targeting implied predicate-argument relationships is the 2010 SemEval task of *Linking Events and Their Participants in Discourse* (Ruppenhofer et al., 2009).

This task extends *Semantic Role Labeling* to cases in which a core argument of a predicate is missing in the syntactic structure but a filler for the corresponding semantic role appears elsewhere and can be inferred from discourse. For example, in the following sentence the semantic role *goal* is unfilled:

(iii) He arrived (*O^{Goal}*) at 8pm.

Yet, we can expect to find an implied filler for *goal* elsewhere in the document.

The SemEval task, termed henceforth as *Implied SRL*, involves three major sub-tasks. First, for each predicate, the unfilled roles, termed *Null Instantiations (NI)*, should be detected. Second, each NI should be classified as *Definite NI (DNI)*, meaning that the role filler must exist in the discourse, or *Indefinite NI* otherwise. Third, the DNI fillers should be found (DNI linking).

Later works that followed the SemEval challenge include (Silberer and Frank, 2012) and (Roth and Frank, 2013), which proposed auto-

matic dataset generation methods and features which capture discourse phenomena. Their highest result was 12% F1-score. Another work is the probabilistic model of Laparra and Rigau (2012), which is trained by properties captured not only from implicit arguments but also from explicit ones, resulting in 19% F1-score. Another notable work is (Gerber and Chai, 2012), which was limited to ten carefully selected nominal predicates.

2.1 Annotations vs. Recognition

Comparing to the implied SRL task, our task may better fit the needs of textual inference. First, some relatively complex steps of the implied SRL task are avoided in our setting, while on the other hand it covers more relevant cases.

More concretely, in textual inference the candidate predicate and argument are typically identified, as they are aligned by the RTE system to a predicate and an argument of the Hypothesis. Thus, the only remaining challenge is to verify that the sought relationship is implied in the text. Therefore, the sub-tasks of identifying and classifying DNIs can be avoided.

On the other hand, in some cases the candidate argument is not a DNI, but is still required in textual inference. One type of such cases are non-core arguments, which cannot be *Definite NIs*. However, textual inference deals with non-core arguments as well (see example 3 in Table 1).

Another case is when an implied predicate-argument relationship holds even though the corresponding role is already filled by another argument, hence not an NI. Consider example 4 of Table 1. While the object of “threatened” is filled (in the Text) by “southeastern United States”, a human reader also infers the “threatened-New Orleans” relationship. Such cases might follow a meronymy relation between the filler (“southeastern United States”) and the candidate argument (“New Orleans”), or certain types of discourse (co)references (e.g., example 5 in Table 1), or some other linguistic phenomena. Either way, they are crucial for textual inference, while not being NIs.

3 Dataset

This section describes a semi-automatic method for extracting candidate instances of implied predicate-argument relationship from an RTE dataset. This extraction process directly follows our task formalization. Given a Text Hypothe-

sis pair, we locate a predicate-argument relationship in the Hypothesis, where both the predicate and the argument appear also in the Text, while the relationship between them is not expressed in its syntactic structure. This process is performed automatically, based on syntactic parsing (see below). Then, a human reader annotates each instance as “Yes” – meaning that the implied relationship indeed holds in the Text, or “No” otherwise. Example instances, constructed by this process, are shown in Table 1.

In this work we used lemma-level lexical matching, as well as nominalization matching, to align the Text predicates and arguments to the Hypothesis. We note that more advanced matching, e.g., by utilizing knowledge resources (like WordNet), can be performed as well. To identify *explicit* predicate-argument relationships we utilized dependency parsing by the Easy-First parser (Goldberg and Elhadad, 2010). Nominalization matching (e.g., example 1 of Table 1) was performed with Nomlex (Macleod et al., 1998).

By applying this method on the RTE-6 dataset (Bentivogli et al., 2010), we constructed a dataset of 4022 instances, where 2271 (56%) are annotated as positive instances, and 1751 as negative ones. This dataset is significantly larger than prior datasets for the implied SRL task. To calculate inter-annotator agreement, the first author also annotated 185 randomly-selected instances. We have reached high agreement score of 0.80 Kappa. The dataset is freely available at www.cs.biu.ac.il/~nlp/resources/downloads/implied-relationships.

4 Recognition Algorithm

We defined 15 features, summarized in Table 2, which capture local and discourse phenomena. These features do not depend on manually built resources, and hence are portable to resource-poor languages. Some features were proposed in prior works, and are marked by G&C (Gerber and Chai, 2012) or S&F (Silberer and Frank, 2012). Our best results were obtained with the *Random Forests* learning algorithm (Breiman, 2001). The first two features are described in the next subsection, while the others are explained in the table itself.

4.1 Statistical discourse features

Statistical features in prior works mostly capture general properties of the predicate and the

#	Category	Feature	Prev. work
1	statistical	co-occurring predicate (explained in subsection 4.1)	<i>New</i>
2	discourse	co-occurring argument (explained in subsection 4.1)	<i>New</i>
3	local discourse	co-reference: whether an explicit argument of p co-refers with a .	<i>New</i>
4		last known location: If the NE of a is “location”, and it is the last location mentioned before p in the document.	<i>New</i>
5		argument prominence: The frequency of the lemma of a in a two-sentence windows of p , relative to all entities in that window.	S&F
6		predicate frequency in document: The frequency of p in the document, relative to all predicates appear in the document.	G&C
7	local candidate properties	statistical argument frequency: The Unigram-model likelihood of a in English documents, calculated from a large corpus.	<i>New</i>
8		definite NP: Whether a is a definite NP	G&C
9		indefinite NP: Whether a is an indefinite NP	G&C
10		quantified predicate: Whether p is quantified (i.e., by expressions like “every ...”, “a good deal of ...”, etc.)	G&C
11		NE mismatch: Whether a is a named entity but the corresponding argument in the hypothesis is not, or vice versa.	<i>New</i>
12	predicate-argument relatedness	predicate-argument frequency: The likelihood of a to be an argument of p (formally: $Pr(a p)$) in a large corpus.	similar feature in G&C
13		sentence distance: The distance between p and a in sentences.	G&C, S&F
14		mention distance: The distance between p and a in entity-mentions.	S&F
15		shared head-predicate: Whether p and a are themselves arguments of another predicate.	G&C

Table 2: Algorithmic features. p and a denote the candidate predicate and argument respectively.

argument, like selectional preferences, lexical similarities, etc. On the contrary, our statistical features follow the intuition that *explicit* predicate-argument relationships in the discourse provide plausible indication that an *implied* relationship holds as well. In our experiments we collected the statistics from Reuters corpus RCV1 (trec.nist.gov/data/reuters/reuters.html), which contains more than 806,000 documents.

We defined two features: *Co-occurring predicate* and *Co-occurring argument*. Let p and a be the candidate predicate and the argument in the text. While they are not connected syntactically, each of them often has an explicit relationships with *other* terms in the text, that might support the sought (implied) relationship between a and p .

More concretely, a is often an *explicit* argument of another predicate p' . For example, example 6 in Table 1 includes the explicit relationship “derailment of train”, which might indicate the implied relationship “crash of train”. Hence p =“crash”, a =“train” and p' =“derailment”. The *Co-occurring predicate* feature estimates the probability that a

document would contain a as an argument of p , given that a appears elsewhere in that document as an argument of p' , based on explicit predicate-argument relationships in a large corpus.

Similarly, the *Co-occurring argument* feature captures cases where p has another *explicit* argument, a' . This is exemplified in example 5 of Table 1, where p =“renew”, a =“Patriot Act” and a' =“law”. Accordingly, the feature quantifies the probability that a document including the relationship p - a' would also include the relationship p - a .

More details about these features can be found in the first author’s Ph.D. thesis at www.cs.biu.ac.il/~nlp/publications/theses/

5 Results

We tested our method in a cross-validation setting, and obtained high result as shown in the first row of Table 3. Since our task and dataset are novel, there is no direct baseline with which we can compare this result. As a reference point we mention the majority class proportion, and also report a configuration in which only features adopted from prior works (G&C and S&F) are utilized. This

Configuration	Accuracy %	Δ %
Full algorithm	81.0	–
Union of prior work	78.0	3.0
Major category (all true)	56.5	24.5
Ablation tests		
no statistical discourse	79.9	1.1
no local discourse	79.3	1.7
no local candidate properties	79.2	1.8
no predicate-argument relatedness	79.7	1.3

Table 3: Accuracy of our method, followed by baselines and ablation tests.

Configuration (input)	Recall	Precision	F1 %
Explicit only	44.6	44.3	44.4
Human annotations	50.9	43.4	46.8
Algorithm recognition	48.5	42.3	45.2

Table 4: RTE-6 Experiment

comparison shows that the contribution of our new features (3%) is meaningful, which is also statistically significant with $p < 0.01$ using *Bootstrap Resampling* test (Koehn, 2004). The high results show that this task is feasible, and its solutions can be adopted as a component in textual inference systems. The positive contribution of each feature category is shown in ablation tests.

An additional experiment tests the contribution of recognizing implied predicate-argument relationships for overall RTE, specifically on the RTE-6 dataset. For the scope of this experiment we developed a simple RTE system, which uses the F1 optimized logistic regression classifier of Jansche (2005) with two features: lexical coverage and predicate-argument relationships coverage. We ran three configurations for the second feature, where in the first only syntactically expressed relationships are used, in the second all the implied relationships, as detected by a human annotator, are added, and in the third only the implied relationships detected by our algorithm are added.

The results, presented in Table 4, first demonstrate the full potential of the implied relationship recognition task to improve textual entailment recognition (Human annotation vs. Explicit only). One third of this potential improvement is achieved by our algorithm¹. Note that all these results are higher than the median result in the RTE-6 challenge (36.14%). While the delta in the F1 score is small in absolute terms, such magnitudes

¹Following the relatively modest size of the RTE dataset, the Algorithm vs. Explicit result is not statistically significant ($p \simeq 0.1$). However, the Human annotation vs. Explicit result is statistically significant with $p < 0.01$.

are typical in RTE for most resources and tools (see (Bentivogli et al., 2010)).

6 Discussion and Conclusions

We formulated the task of recognizing implied predicate-argument relationships within textual inference scenarios. We compared this task to the labeling task of SemEval 2010, where no prior information about candidate arguments in the text is available. We point out that in textual inference scenarios the candidate predicate and argument are given by the Hypothesis, while the challenge is only to verify that a predicate-argument relationship between these candidates is implied from the given Text. Accordingly, some complex steps necessitated in the SemEval task can be avoided, while additional relevant cases are covered.

Moreover, we have shown that this simpler task is more feasibly solvable, where our 15 features achieved more than 80% accuracy.

While our dataset and algorithm were presented in the context of RTE, the same challenge and methods are applicable to other textual inference tasks as well. Consider, for example, the *Question Answering (QA)* task. Typically QA systems detect a candidate predicate that matches the question’s predicate. Similarly, candidate arguments, which match either the expected answer type or other arguments in the question are detected too. Consequently, our methods which exploit the availability of the candidate predicate and argument can be adapted to this scenario as well.

Similarly, a typical approach for *Event Extraction* (a sub task of *Information Extraction*) is to start by applying an entity extractor, which identifies argument candidates. Accordingly, candidate predicate and arguments are detected in this scenario too, while the remaining challenge is to assess the likelihood that a predicate-argument relationship holds between them.

Following this observation, we propose future work of applying our methods to other tasks. An additional direction for future work is to further develop new methods for our task, possibly by incorporating SRL resources and/or linguistically oriented rules, in order to improve the results we achieved so far.

Acknowledgments

This work was partially supported by the EC-funded project EXCITEMENT (FP7ICT-287923).

References

- Luisa Bentivogli, Peter Clark, Ido Dagan, Hoa Trang Dang, and Danilo Giampiccolo. 2010. The sixth pascal recognizing textual entailment challenge. In *Proceedings of TAC*.
- Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1).
- Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment*, pages 177–190.
- Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The stanford typed dependencies representation. In *proceedings of COLING 2008 Workshop on Cross-framework and Cross-domain Parser Evaluation*.
- Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of ACL*.
- Matthew Gerber and Joyce Y. Chai. 2012. Semantic role labeling of implicit arguments for nominal predicates. *Computational Linguistics*.
- Yoav Goldberg and Michael Elhadad. 2010. An efficient algorithm for easy-first non-directional dependency parsing. In *Proceedings of NAACL*.
- Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of EMNLP*.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. *SIGKDD Explorations*, 11(1).
- Martin Jansche. 2005. Maximum expected f-measure training of logistic regression models. In *Proceedings of EMNLP*.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*.
- Egoitz Laparra and German Rigau. 2012. Exploiting explicit annotations and semantic types for implicit argument resolution. In *Proceedings of IEEE-ICSC*.
- Catherine Macleod, Ralph Grishman, Adam Meyers, Leslie Barrett, and Ruth Reeves. 1998. Nomlex: A lexicon of nominalizations. In *Proceedings of EU-RALEX*.
- Michael Roth and Anette Frank. 2013. Automatically identifying implicit arguments to improve argument linking and coherence modeling. In *Proceedings of *SEM*.
- Josef Ruppenhofer, Caroline Sporleder, Roser Morante, Collin Baker, and Martha Palmer. 2009. Semeval-2010 task 10: Linking events and their participants in discourse. In *The NAACL-HLT 2009 Workshop on Semantic Evaluations: Recent Achievements and Future Directions (SEW-09)*.
- Josef Ruppenhofer, Caroline Sporleder, Roser Morante, Collin Baker, and Martha Palmer. 2010. Semeval-2010 task 10: Linking events and their participants in discourse. In *Proceedings of the 5th International Workshop on Semantic Evaluation*.
- Carina Silberer and Anette Frank. 2012. Casting implicit role linking as an anaphora resolution task. In *Proceedings of *SEM*.
- Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of NAACL*.

Measuring metaphoricity

Jonathan Dunn

Department of Computer Science / Illinois Institute of Technology

jonathan.edwin.dunn@gmail.com

Abstract

This paper presents the first computationally-derived scalar measurement of metaphoricity. Each input sentence is given a value between 0 and 1 which represents how metaphoric that sentence is. This measure achieves a correlation of 0.450 (Pearson's R , $p < 0.01$) with an experimental measure of metaphoricity involving human participants. While far from perfect, this scalar measure of metaphoricity allows different thresholds for metaphoricity so that metaphor identification can be fitted for specific tasks and datasets. When reduced to a binary classification evaluation using the VU Amsterdam Metaphor Corpus, the system achieves an F-Measure of 0.608, slightly lower than the comparable binary classification system's 0.638 and competitive with existing approaches.

1 Introduction

Metaphor is a cognitive phenomenon (Lakoff & Johnson, 1980, 1999) which has a significant impact on human reasoning abilities (Casasanto & Jasmin, 2012; Johansson Falk & Gibbs, 2012) and which, as a result, commonly appears in language in the form of metaphoric expressions (e.g., Deignan, 2005). The most comprehensive non-computational study of metaphoric expressions in large corpora (Steen, et al., 2010) found that up to 18.5% of words in the British National Corpus were used metaphorically. This means that metaphorically used words not only have very different interpretations than literally used words, but they are also common enough to pose a significant challenge for computational linguistics.

Starting with Wilks (1978), the problem of metaphor has been approached as an identifica-

tion task: first identify or detect metaphoric expressions and then (1) prevent them from interfering with computational treatments of literal expressions and (2) use them to gain additional insight about a text (e.g., Carbonell, 1980; Neuman & Nave, 2009). The identification or detection task has been approached as a binary classification problem: for a given unit of language (e.g., word, phrase, sentence) decide whether it is metaphoric or non-metaphoric. Wilks (1978) used selectional restrictions for this purpose; Mason (2004) used hand-crafted knowledge resources to detect similar selectional mismatches; another approach is to detect selectional mismatches using statistically created resources (e.g., Shutova, et al. 2013; Shutova & Sun, 2013). A second general approach to the binary classification problem has been to use mismatches in properties like abstractness (Gandy, et al., 2013; Assaf, et al., 2013; Tsvetkov, et al., 2013; Turney, et al., 2011), semantic similarity (Li & Sporleder, 2010; Sporleder & Li, 2010), and domain membership (Dunn, 2013a, 2013b) to identify metaphoric units of language. A third approach has been to use forms of topic modelling to identify linguistic units which represent both a metaphoric topic and a literal topic (Strzalkowski, 2013; Bracewell, et al, 2013; Mohler, et al., 2013).

The single constant across all of these approaches is that the task is viewed as a binary classification problem of distinguishing metaphoric language from non-metaphoric language. This binary distinction assumes a clear boundary between the two; in other words, it assumes that metaphoricity is a discrete property. However, three strands of theoretical research show that metaphoricity is not a discrete property. First, psycholinguistic studies of metaphor processing show that there is no difference between the processing of metaphoric and non-metaphoric language (Coulson & Matlock, 2001; Gibbs, 2002; Evans, 2010). The most plausible interpretation

of this psycholinguistic evidence is that most linguistic units fall somewhere between metaphoric and literal, so that metaphoricity is a scalar value which influences processing gradually (and is difficult to uncover because of related factors like salience; Giora, 2002). Second, linguistic studies of metaphor have found that the metaphoricity of a linguistic unit can be predicted given certain factors (Dunn, 2011, 2013c). Third, the high frequency of metaphorically used language implies that it is hard to set a boundary beyond which a word is used metaphorically. In other words, it seems clear that 18.5% of the BNC is not highly metaphoric, but rather is the sort of slightly metaphoric language that speakers are not consciously aware of because it is used so frequently.

This paper introduces a system for producing a scalar measurement of metaphoricity which places sentences anywhere between 0 (literal) and 1 (highly metaphoric). The goal is to produce a computationally derived measurement that models the gradient nature of metaphoricity, with the result that metaphors which are clearly and consciously seen as metaphors score closer to 1 and metaphors which are not realized by speakers to be metaphoric score further from 1. This scalar measurement approach has two advantages: (1) it adheres more closely to the current theoretical understanding of metaphor, thus being more cognitively accurate; (2) it allows applications to control the threshold of metaphoricity when identifying metaphor, thus allowing the treatment of metaphor to be optimized for a given task.

2 Measuring Gradient Metaphoricity

An experiment was conducted to set a standard for evaluating scalar measurements of metaphoricity. A corpus of 60 sentences of varying metaphoricity, drawn equally from four top-level domains (PHYSICAL, MENTAL, SOCIAL, and ABSTRACT), was created using the Corpus of Contemporary American English. Each domain was represented by five verbs and each verb by three sentences: one literal, one slightly metaphoric, and one very metaphoric (as judged by the author).

The selection of various domains, verbs, and hypothesized metaphoricity levels helps to control for other factors, like abstractness, which might be only indirectly related to metaphoricity. It also ensures that the experiment covers a wide-range of metaphors. It should be noted that the purpose

of the experiment is not to (1) test a three-way distinction between metaphoricity levels (which is simply used to ensure a representative selection of metaphors) or (2) test the author's intuitions of metaphoricity. Rather, the purpose is to have a representative selection of metaphors rated for metaphoricity against which to test scalar measurements of metaphoricity.

Three survey tasks were used. The first tested speakers' ability to consistently separate metaphoric and non-metaphoric sentences. Participants were given a sentence and asked to identify it as "Literal" or "Metaphoric." The second task tested speakers' ability to consistently label a given sentence as "Not Metaphoric", "Slightly Metaphoric", and "Very Metaphoric." The additional label was added in order to provide participants with a middle ground between metaphoric and literal. The third task tested speakers' ability to consistently rank three sentences according to their metaphoricity. In order to ensure comparability, each set of three sentences contained a literal, a slightly metaphoric, and a very metaphoric use of a single verb (e.g., three uses of "butcher"). The purpose of this task was to allow participants to directly compare different uses of the same verb.

The surveys were conducted using the MechanicalTurk platform. Each participant took a particular survey only once and the sentences to be rated were drawn randomly from the corpus. Participants were given eight questions for the identification and labeling tasks and four questions for the ranking task. This was done in order to keep the survey short and prevent participants from losing interest. All participants were asked if they had attended a primary or elementary school conducted in English in order to ensure consistent language ability. Further, a test question was positioned part way through the survey to ensure that participants read the prompts correctly. Only answers valid according to these two tests are considered in the following results. Each task had 100 unique participants who gave valid answers, for a total of 300 participants. Participants did not see any domain information for the sentence prompts.

For the first task, the binary identification task, the metaphoricity of a sentence was computed by taking the percentage of participants who identified it as metaphoric. Thus, if all participants agreed that a sentence was metaphoric, then it receives a 1, while if half of the participants agreed,

then it receives a 0.5. The idea here is that high metaphoricity is consciously available to participants, so that the more agreement there is about metaphor the more the participants are aware of the sentence's metaphoricity and thus the higher its metaphoricity value should be. The results of this first experiment are summarized in Table 1 with the mean, standard deviation, and range of the metaphoricity measurements. The results are strong on the low end of the scale, with every domain having sentences with either 0 values or close to 0 values. The high end is more problematic, with the highest values in each domain being below 0.9. This is a result of not having perfect agreement across all participants. However, in spite of this, the measure makes a good distinction between utterances. For example, it assigns the metaphoricity value of 0.833 to the sentence in (1), but a metaphoricity value of only 0.153 to the sentence in (2). This reflects a distinction in metaphoricity, although the extreme top and bottom of the scale are problematic.

(1) "A lady on high heels clacked along, the type my mother says invests all of her brainpower in her looks."

(2) "The banks and the corporations in America today have lots of money that they can invest right now."

Domain	Mean	Std. Dev.	Range
Abstract	0.373	0.282	0.065–0.833
Mental	0.289	0.278	0.000–0.888
Physical	0.417	0.331	0.000–0.846
Social	0.389	0.351	0.000–0.812
All	0.367	0.316	0.000–0.888

Table 1: Metaphoricity by identification.

The second experiment asks participants to label metaphoricity, this time including a distinction between slightly metaphoric and highly metaphoric sentences. The purpose of this is not to test a three-way distinction in metaphoricity values, but rather to improve the scale by moving intermediate sentences out of the Literal or Metaphoric categories. The metaphoricity values for this experiment were calculated in the same way: the percentage of participants who rated a sentence as highly metaphoric. Thus, this measurement also is based on the idea that more participants will be consciously aware of highly metaphoric sentences, with a third category avail-

able to allow an extra distinction to be made. This measurement, summarized in Table 2, is more accurate at the lower end of the scale, with many sentences receiving a 0 because participants were able to choose a category other than metaphoric. At the same time, the values tend to be further from 1 at the upper end of the scale. The sentence in (2) above, for example, received a 0; however, the sentence in (1) above received only a 0.571, which, while high given the range of values, is still far from 1. Thus, while the measurement makes distinctions at the top of the scale, it does not approach 1.

Domain	Mean	Std. Dev.	Range
Abstract	0.170	0.165	0.000–0.571
Mental	0.096	0.119	0.000–0.455
Physical	0.220	0.248	0.000–0.778
Social	0.258	0.281	0.000–0.769
All	0.186	0.222	0.000–0.778

Table 2: Metaphoricity by labelling.

The third task gathered ordering information by presenting participants with three sentences, all of which contained the same main verb. The participants were asked to order the sentences from the least metaphoric to the most metaphoric. The purpose of this experiment was to give participants context in the form of other uses of a given verb against which to make their judgments. The metaphoricity value was computed by taking the percentage of participants who identified a sentence as the most metaphoric of the three given sentences. This measurement, shown in Table 3, has similar averages across domains, unlike the previous measurements. It tends to be better than the previous measures on the upper bound, likely because of the contextual comparison it allows. However, because sentences with the same main verb were forced into a three-way ordering, participants could not, for example, label two of the sentences as equally metaphoric. Thus, it is possible that some of this advantage on the upper bound is a result of the task itself.

Given these three experiments for measuring the metaphoricity of sentences, Table 4 shows the correlations between each measure using Pearson's R. Each correlation is significant at the 0.01 level (2-tailed). The highest correlation is between the first and second tasks, at 0.819. The lowest is between the first and third (which differ in the

Domain	Mean	Std. Dev.	Range
Abstract	0.333	0.211	0.056–0.773
Mental	0.331	0.175	0.071–0.632
Physical	0.331	0.235	0.050–0.941
Social	0.327	0.280	0.050–0.783
All	0.331	0.227	0.050–0.941

Table 3: Metaphoricity by ordering.

number of distinctions allowed) at 0.699. However, this is still a high correlation.

Task	Identify	Label	Order
Identify	–	0.819	0.699
Label	0.819	–	0.702
Order	0.699	0.702	–

Table 4: Correlation between measurements.

This section has put forward a robust series of scalar measurements of metaphoricity. Each experiment had 100 participants and operationalized the task of rating metaphoricity in different ways across a representative section of domains, verbs, and metaphoricity levels. The resulting highly correlated measures show that we have a good standard of metaphoricity against which to evaluate computational models which produce scalar measurements of metaphoricity. The next section introduces such a system.

3 Description of the System

We approach the problem by starting with an existing binary identification system and converting it to a scalar system. In principle any of the property-based systems listed above could be converted in this way. We have chosen to start with the domain interaction system (Dunn, 2013a, 2013b), which performed competitively in an evaluation with other systems (Dunn, 2013b). The original system uses the properties of domain-membership and event-status of concepts to identify metaphors at the sentence-level using a logistic regression classifier. The scalar version of the system will have to evaluate the features in a different way.

The first step is to increase the robustness of the system’s representation of sentences by adding additional properties. We split the original system’s domain membership feature into two: the domain of a word’s referent and the domain of a word’s sense. The idea is to capture cases like MINISTER,

in which a physical object (a human) is defined by its social role (being a minister). The event-status property is unchanged.

Several additional properties are added; these properties were not used in the original system. First, animacy-status allows a distinction to be made between inanimate objects like rocks and stones and animate or human objects. Second, the fact-status property allows a distinction to be made between objects which exist as such independently of humans (e.g., rocks and stones) and those which exist to some degree dependent on human consciousness (e.g., laws and ideas). Third, the function-status property allows a distinction to be made between objects which encode a function (e.g., a screwdriver is specifically an object meant to turn screws) and those which do not encode a function (e.g., rocks are simply objects). A finer distinction within the function-status property distinguishes social functions (e.g., laws) from physical-use functions (e.g., screwdrivers).

Following the original system, these properties are taken from a knowledge-base and used to create feature vectors. The text is first processed using Apache OpenNLP for tokenization, named entity recognition, and part of speech tagging. Morpha (Minnen, et al., 2001) is used for lemmatization. At this point word sense disambiguation is performed using SenseRelate (Pedersen & Kolhatkar, 2009), mapping the lexical words to the corresponding WordNet senses. These WordNet senses are first mapped to SynSets and then to concepts in the SUMO ontology, using existing mappings (Niles & Pease, 2001, 2003).

Thus, each sentence is represented by the SUMO concepts which it contains and each concept is represented by its six concept properties. The features used are computed as follows: First, the relative frequency of each value of each concept property in the sentence is determined; Second, the number of instances of the most common value for each property is determined, as well as the number of instances of all other values (both relativized to the number of concepts present in the sentence). Third, the number of types of values for each concept property is determined, relative to the number of possible types. This gives a total of 41 features for each sentence.

These features were computed for each of the sentences used in the experiments and then

the correlation between the features and the metaphoricity measurements were computed using Pearson’s R. Those features which had a significant positive relationship with the experimental results, shown in Table 5, were added together to create a rough computational measure of metaphoricity and then converted so that they fall between 0 and 1. The resulting computationally-derived measure correlates significantly with each of the experiments: 0.450, 0.416, and 0.337.

Properties	Values
Domain of the Referent	Mental
Domain of the Referent	Other / Concepts
Event-Status	State
Animacy-Status	Undetermined
Animacy-Status	Other / Concepts
Fact-Status	Physical
Function-Status	None
Domain of the Referent	Types / Possible
Event-Status	Types / Possible
Animacy-Status	Types / Possible
Function-Status (negative)	Types / Possible

Table 5: Predictive features.

4 Evaluation

A scalar measurement of metaphoricity allows the threshold for metaphor in metaphor identification tasks to be fitted for specific purposes and datasets. The scalar system was evaluated on the VU Amsterdam Metaphor Corpus (Steen, et al., 2010) which consists of 200,000 words from the British National Corpus divided into four genres (academic, news, fiction, and spoken; performance on the spoken genre was not evaluated for this task because it consists of many short fragmentary utterances) and manually annotated for metaphor by five raters. Previous evaluations using this corpus (Dunn, 2013b) concluded that prepositions annotated as metaphoric in the corpus should not be considered metaphoric for computational purposes. Thus, metaphorically used prepositions have been untagged as metaphoric. Further, we have also untagged the ambiguously metaphoric sentences. Sentences with an insufficiently robust conceptual representation were removed (e.g., fragments). The evaluation dataset thus consists of 6,893 sentences, distributed as shown in Table 6.

For the purposes of this evaluation, the thresh-

Subset	Literal	Metaphor	Total
Academic	759	1,550	2,309
Fiction	1,215	1,389	2,604
News	366	1,614	1,980
Total	2,340	4,553	6,893

Table 6: Size of evaluation dataset in sentences.

old for metaphor was set independently for each genre and tied to the number of sentences containing metaphorically used words, as rated by the annotators of the corpus. Thus, for the number x of metaphors in the genre, the x sentences with the top metaphoricity values were identified as metaphoric. This illustrates the flexibility of such a scalar approach to metaphor identification. The baseline results are taken from a binary classification evaluation of the corpus using the full set of 41 features produced by the system and evaluated using the logistic regression algorithm with 100-fold cross-validation.

System	Subset	Prec.	Recall	F-Meas.
Scalar	Acad.	0.578	0.686	0.578
Binary	Acad.	0.649	0.682	0.623
Scalar	News	0.712	0.822	0.712
Binary	News	0.750	0.812	0.748
Scalar	Fict.	0.554	0.582	0.554
Binary	Fict.	0.632	0.633	0.630
Scalar	All	0.608	0.703	0.608
Binary	All	0.663	0.685	0.638

Table 7: Evaluation results.

The binary classification system, with access to the full range of features, out-performs the scalar measurement in most cases. It is important to note, however, that the binary classification system requires labelled training data and is restricted to a single threshold of metaphoricity, in this case the threshold embedded in the corpus by the raters. The scalar system, however, was trained only on the experimental data and was not influenced by the evaluation corpus (except, of course, that it had access to the number of metaphoric sentences in the dataset, which is a parameter and not part of the model itself). Further, it can be applied to any English text without the need for labelled training data. Thus, the scalar approach performs competitively on a binary task (0.608 vs. 0.638 F-Measure) but can also produce scalar identifications, which binary systems cannot produce.

References

- Assaf, D., Neuman, Y., Cohen, Y., Argamon, S., Howard, N., Last, M., Koppel, M. 2013. Why “dark thoughts” aren’t really dark: A novel algorithm for metaphor identification. *2013 IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain*: 60–65. Institute of Electrical and Electronics Engineers.
- Bracewell, D. B., Tomlinson, M. T., Mohler, M. 2013. Determining the Conceptual Space of Metaphoric Expressions. *Proceedings of the 14th International Conference on Computational Linguistics and Intelligent Text Processing, Volume I*: 487–500. Berlin, Heidelberg: Springer-Verlag.
- Carbonell, J. 1980. Metaphor - A Key to Extensible Semantic Analysis. *Proceedings of the 18th Meeting of the Association for Computational Linguistics*: 17–21. Association for Computational Linguistics.
- Casasanto, D., Jasmin, K. 2012. The Hands of Time: Temporal gestures in English speakers. *Cognitive Linguistics*, 23(4): 643–674.
- Coulson, S., Matlock, T. 2001. Metaphor and the space structuring model. *Metaphor & Symbol*, 16(3), 295-316.
- Deignan, A. 2005. *Metaphor and Corpus Linguistics*. Amsterdam: John Benjamins.
- Dunn, J. 2011. Gradient Semantic Intuitions of Metaphoric Expressions. *Metaphor & Symbol*, 26(1), 53-67.
- Dunn, J. 2013a. Evaluating the premises and results of four metaphor identification systems. *Proceedings of the 14th International Conference on Computational Linguistics and Intelligent Text Processing, Volume I*: 471-486. Berlin, Heidelberg: Springer-Verlag.
- Dunn, J. 2013b. What metaphor identification systems can tell us about metaphor-in-language. *Proceedings of the First Workshop on Metaphor in NLP*: 1-10. Association for Computational Linguistics.
- Dunn, J. 2013c. How linguistic structure influences and helps to predict metaphoric meaning. *Cognitive Linguistics*, 24(1), 33-66.
- Evans, V. 2010. Figurative language understanding in LCCM Theory. *Cognitive Linguistics*, 21(4), 601-662.
- Gandy, L., Allan, N., Atallah, M., Frieder, O., Howard, N., Kanareykin, S., Argamon, S. 2013. Automatic Identification of Conceptual Metaphors With Limited Knowledge. *Proceedings of the 27th Conference on Artificial Intelligence*: 328–334. Association for the Advancement of Artificial Intelligence.
- Gibbs Jr., R. W. 2002. A new look at literal meaning in understanding what is said and implicated. *Journal of Pragmatics*, 34(4), 457-486.
- Giora, R. 2002. Literal vs. figurative language: Different or equal? *Journal of Pragmatics*, 34(4), 487-506.
- Johansson Falck, M., Gibbs, Jr., R. W. 2012. Embodied motivations for metaphorical meanings. *Cognitive Linguistics*, 23(2): 251–272.
- Lakoff, G., Johnson, M. 1980. *Metaphors we live by*. Chicago: University Of Chicago Press.
- Lakoff, G., Johnson, M. 1999. *Philosophy in the flesh: The embodied mind and its challenge to western thought*. Chicago: University Of Chicago Press.
- Li, L., Sporleder, C. 2010a. Linguistic Cues for Distinguishing Literal and Non-literal Usages. *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*: 683-691. Association for Computational Linguistics.
- Li, L., Sporleder, C. 2010b. Using Gaussian Mixture Models to Detect Figurative Language in Context. *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*: 297–300. Association for Computational Linguistics.
- Mason, Z. 2004. CorMet: A Computational, Corpus-Based Conventional Metaphor Extraction System. *Computational Linguistics*, 30(1), 23-44.
- Minnen, G., Carroll, J., Pearce, D. 2001. Applied morphological processing of English. *Natural Language Engineering*, 7(3), 207-223.
- Mohler, M., Bracewell, D., Tomlinson, M., Hinote, D. 2013. Semantic Signatures for Example-Based Linguistic Metaphor Detection. *Proceedings of the First Workshop on Metaphor in NLP*: 27-35. Association for Computational Linguistics.
- Neuman, Y., Nave, O. 2009. Metaphor-based meaning excavation. *Information Sciences*, 179, 2719-2728.
- Niles, I., Pease, A. 2001. Towards a standard upper ontology. *Proceedings of the International Conference on Formal Ontology in Information Systems*: 2-9. Association for Computing Machinery.
- Niles, I., Pease, A. 2003. Linking lexicons and ontologies: Mapping WordNet to the Suggested Upper Merged Ontology. *Proceedings of the 2003 International Conference on Information and Knowledge Engineering*: 412-416. World Congress in Computer Science, Computer Engineering, and Applied Computing.
- Pedersen, T., Kolhatkar, V. 2009. WordNet::SenseRelate::AllWords - A broad coverage word sense tagger that maximizes semantic relatedness. *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North*

American Chapter of the Association for Computational Linguistics, Companion Volume: Demonstration Session: 17-20. Association for Computational Linguistics.

- Shutova, E., Sun, L. 2013. Unsupervised Metaphor Identification using Hierarchical Graph Factorization Clustering. *Proceedings of Human Language Technologies: The 2013 Annual Conference of the North American Chapter of the Association for Computational Linguistics: 978-988.* Association for Computational Linguistics.
- Shutova, E., Teufel, S., Korhonen, A. 2013. Statistical Metaphor Processing. *Computational Linguistics*, 39(2), 301-353.
- Steen, G. J., Dorst, A. G., Herrmann, J. B., Kaal, A. A., Krennmayr, T. 2010. Metaphor in usage. *Cognitive Linguistics*, 21(4), 765-796.
- Strzalkowski, T., Broadwell, G. A., Taylor, S., Feldman, L., Shaikh, S., Liu, T., Elliot, K. 2013. Robust Extraction of Metaphor from Novel Data. *Proceedings of the First Workshop on Metaphor in NLP: 67-76.* Association for Computational Linguistics.
- Tsvetkov, Y., Mukomel, E., Gershman, A. 2013. Cross-Lingual Metaphor Detection Using Common Semantic Features. *Proceedings of the First Workshop on Metaphor in NLP: 45-51.* Association for Computational Linguistics.
- Turney, P. D., Neuman, Y., Assaf, D., Cohen, Y. 2011. Literal and Metaphorical Sense Identification Through Concrete and Abstract Context. *Proceedings of the Conference on Empirical Methods in Natural Language Processing: 680-690.* Association for Computational Linguistics.
- Wilks, Y. 1978. Making preferences more active. *Artificial Intelligence*, 11(3), 197-223.

Empirical Study of Unsupervised Chinese Word Segmentation Methods for SMT on Large-scale Corpora

Xiaolin Wang Masao Utiyama Andrew Finch Eiichiro Sumita

National Institute of Information and Communications Technology

{xiaolin.wang,mutiyama,andrew.finch,eiichiro.sumita}@nict.go.jp

Abstract

Unsupervised word segmentation (UWS) can provide domain-adaptive segmentation for statistical machine translation (SMT) without annotated data, and bilingual UWS can even optimize segmentation for alignment. Monolingual UWS approaches of explicitly modeling the probabilities of words through Dirichlet process (DP) models or Pitman-Yor process (PYP) models have achieved high accuracy, but their bilingual counterparts have only been carried out on small corpora such as basic travel expression corpus (BTEC) due to the computational complexity. This paper proposes an efficient unified PYP-based monolingual and bilingual UWS method. Experimental results show that the proposed method is comparable to supervised segmenters on the in-domain NIST OpenMT corpus, and yields a 0.96 BLEU relative increase on NTCIR PatentMT corpus which is out-of-domain.

1 Introduction

Many languages, especially Asian languages such as Chinese, Japanese and Myanmar, have no explicit word boundaries, thus word segmentation (WS), that is, segmenting the continuous texts of these languages into isolated words, is a prerequisite for many natural language processing applications including SMT.

Though supervised-learning approaches which involve training segmenters on manually segmented corpora are widely used (Chang et al., 2008), yet the criteria for manually annotating words are arbitrary, and the available annotated corpora are limited in both quantity and genre variety. For example, in machine translation, there are various parallel corpora such as

BTEC for tourism-related dialogues (Paul, 2008) and PatentMT in the patent domain (Goto et al., 2011)¹, but researchers working on Chinese-related tasks often use the Stanford Chinese segmenter (Tseng et al., 2005) which is trained on a small amount of annotated news text.

In contrast, UWS, spurred by the findings that infants are able to use statistical cues to determine word boundaries (Saffran et al., 1996), relies on statistical criteria instead of manually crafted standards. UWS learns from unsegmented raw text, which are available in large quantities, and thus it has the potential to provide more accurate and adaptive segmentation than supervised approaches with less development effort being required.

The approaches of explicitly modeling the probability of words (Brent, 1999; Venkataraman, 2001; Goldwater et al., 2006; Goldwater et al., 2009; Mochihashi et al., 2009) significantly outperformed a heuristic approach (Zhao and Kit, 2008) on the monolingual Chinese SIGHAN-MSR corpus (Emerson, 2005), which inspired the work of this paper.

However, bilingual approaches that model word probabilities suffer from computational complexity. Xu et al. (2008) proposed a bilingual method by adding alignment into the generative model, but was only able to test it on small-scale BTEC data. Nguyen et al. (2010) used the local best alignment to increase the speed of the Gibbs sampling in training but the impact on accuracy was not explored.

This paper is dedicated to bilingual UWS on large-scale corpora to support SMT. To this end, we model bilingual UWS under a similar framework with monolingual UWS in order to improve efficiency, and replace Gibbs sampling with expectation maximization (EM) in training.

We aware that variational bayes (VB) may be used for speeding up the training of DP-based

¹<http://ntcir.nii.ac.jp/PatentMT>

or PYP-based bilingual UWS. However, VB requires formulating the m expectations of $(m - 1)$ -dimensional marginal distributions, where m is the number of hidden variables. For UWS, the hidden variables are indicators that identify substrings of sentences in the corpus as words. These variables are large in number and it is not clear how to apply VB to UWS, and as far the authors aware there is no previous work related to the application of VB to monolingual UWS. Therefore, we have not explored VB methods in this paper, but we do show that our method is superior to the existing methods.

The contributions of this paper include,

- state-of-the-art accuracy in monolingual UWS;
- the first bilingual UWS method practical for large corpora;
- improvement of BLEU scores compared to supervised Stanford Chinese word segmenter.

2 Methods

This section describes our unified monolingual and bilingual UWS scheme. Table 1 lists the main notation. The set \mathcal{F} is chosen to represent an unsegmented foreign language sentence (a sequence of characters), because an unsegmented sentence can be seen as the set of all possible segmentations of the sentence denoted F , i.e. $F \in \mathcal{F}$.

Notation	Meaning
\mathcal{F}	an unsegmented foreign sentence
$\mathcal{F}_k^{k'}$	unsegmented substring of the underlying string of \mathcal{F} from k to k'
F	a segmented foreign sentence
f_j	the j -th foreign word
\mathcal{M}	monolingual segmentation model
$P_{\mathcal{M}}(x)$	probability of x being a word according to M
E	a tokenized English sentence
e_i	the i -th English word
(\mathcal{F}, E)	a bilingual sentence pair
\mathcal{B}	bilingual segmentation model
$P_{\mathcal{B}}(x e_i)$	probability of x being a word according to B given e_i

Table 1: Main Notation.

Monolingual and bilingual WS can be formulated as follows, respectively,

$$\hat{F}(\mathcal{F}) = \underset{F \in \mathcal{F}}{\operatorname{argmax}} P(F|\mathcal{F}, \mathcal{M}), \quad (1)$$

$$\hat{F}(\mathcal{F}, E) = \underset{F \in \mathcal{F}}{\operatorname{argmax}} \sum_a P(F, a|\mathcal{F}, E, \mathcal{B}), \quad (2)$$

where a is an alignment between F and E . The English sentence E is used in the generation of a segmented sentence F .

UWS learns models by maximizing the likelihood of the unsegmented corpus, formulated as,

$$\hat{\mathcal{M}} = \underset{\mathcal{M}}{\operatorname{argmax}} \prod_{\mathcal{F} \in \mathbb{F}} \left(\sum_{F \in \mathcal{F}} P(F|\mathcal{M}) \right), \quad (3)$$

$$\hat{\mathcal{B}} = \underset{\mathcal{B}}{\operatorname{argmax}} \prod_{(\mathcal{F}, E) \in \mathbb{B}} \left(\sum_{F \in \mathcal{F}} \sum_a P(F, a|\mathcal{F}, E, \mathcal{B}) \right). \quad (4)$$

Our method of learning \mathcal{M} and \mathcal{B} proceeds in a similar manner to the EM algorithm. The following two operations are performed iteratively for each sentence (pair).

- Exclude the previous expected counts of the current sentence (pair) from the model, and then derive the current sentence in all possible ways, calculating the new expected counts for the words (see Section 2.1), that is, we calculate the expected probabilities of the $\mathcal{F}_k^{k'}$ being words given the data excluding \mathcal{F} , i.e. $\mathbf{E}_{\mathbb{F}/\{\mathcal{F}\}}(P(\mathcal{F}_k^{k'}|\mathcal{F})) = P(\mathcal{F}_k^{k'}|\mathcal{F}, \mathcal{M})$ in a similar manner to the marginalization in the Gibbs sampling process which we are replacing;
- Update the respective model \mathcal{M} or \mathcal{B} according to these expectations (see Section 2.2).

2.1 Expectation

2.1.1 Monolingual Expectation

$P(\mathcal{F}_k^{k'}|\mathcal{F}, \mathcal{M})$ is the marginal probability of all the possible $F \in \mathcal{F}$ that contain $\mathcal{F}_k^{k'}$ as a word, which can be calculated efficiently through dynamic programming (the process is similar to the forward-backward algorithm in training a hidden Markov model (HMM) (Rabiner, 1989)):

$$P_a(k) = \sum_{u=1}^U P_a(k-u)P_{\mathcal{M}}(\mathcal{F}_{k-u}^k)$$

$$P_b(k') = \sum_{u=1}^U P_b(k'+u)P_{\mathcal{M}}(\mathcal{F}_{k'+u}^{k'})$$

$$P(\mathcal{F}_k^{k'}|\mathcal{F}, \mathcal{M}) = P_a(k)P_{\mathcal{M}}(\mathcal{F}_k^{k'})P_b(k'), \quad (5)$$

3 Complexity Analysis

The computational complexity of our method is linear in the number of iterations, the size of the corpus, and the complexity of calculating the expectations on each sentence or sentence pair. In practical applications, the size of the corpus is fixed, and we found empirically that the number of iterations required by the proposed method for convergence is usually small (less than five iterations). We now look in more detail at the complexity of the expectation calculation in monolingual and bilingual models.

The monolingual expectation is calculated according to Eq. 5; the complexity is linear in the length of sentences and the square of the predefined maximum length of words. Thus its overall complexity is

$$O_{\text{monoling}}^{\text{unigram}} = O(N_i |F| KU^2), \quad (15)$$

where N_i is the number of iterations, K is the average number of characters per sentence, and U is the predefined maximum length of words.

For the monolingual bigram model, the number of states in the HMM is U times more than that of the monolingual unigram model, as the states at specific position of F are not only related to the length of the current word, but also related to the length of the word before it. Thus its complexity is U^2 times the unigram model's complexity:

$$O_{\text{monoling}}^{\text{bigram}} = O(N_i |F| KU^4). \quad (16)$$

The bilingual expectation is given by Eq. 8, whose complexity is the same as the monolingual case. However, the complexity of calculating the transition probability, in Eqs. 9 and 10, is $O(\delta_b)$. Thus its overall complexity is:

$$O_{\text{biling}}^{\text{unigram}} = O(N_i |F| KU^2 \delta_b). \quad (17)$$

4 Experiments

In this section, the proposed method is first validated on monolingual segmentation tasks, and then evaluated in the context of SMT to study whether the translation quality, measured by BLEU, can be improved.

4.1 Experimental Settings

4.1.1 Experimental Corpora

Two monolingual corpora and two bilingual corpora are used (Table 2). CHILDES (MacWhinney and Snow, 1985) is the most common test

Corpus	Type	# Sentences	# Characters
CHILDES	Mono.	9,790	95,809
SIGHAN-MSR	Mono.	90,903	4,234,824
OpenMT06	Biling.	437,004	19,692,605
PatentMT9	Biling.	1,004,000	63,130,757

Table 2: Experimental Corpora

corpus for UWS methods. The SIGHAN-MSR corpus (Emerson, 2005) consists of manually segmented simplified Chinese news text, released in the SIGHAN bakeoff 2005 shared tasks.

The first bilingual corpus: OpenMT06 was used in the NIST open machine translation 2006 Evaluation ². We removed the United Nations corpus and the traditional Chinese data sets from the constraint training resources. The data sets of NIST Eval 2002 to 2005 were used as the development for MERT tuning (Och, 2003). This data set mainly consists of news text ³. PatentMT9 is from the shared task of NTCIR-9 patent machine translation. The training set consists of 1 million parallel sentences extracted from patent documents, and the development set and test set both consist of 2000 sentences.

4.1.2 Performance Measurement and Baseline Methods

For the monolingual tasks, the F_1 score against the gold annotation is adopted to measure the accuracy. The results reported in related papers are listed for comparison.

For the bilingual tasks, the publicly available system of Moses (Koehn et al., 2007) with default settings is employed to perform machine translation, and BLEU (Papineni et al., 2002) was used to evaluate the quality. Character-based segmentation, LDC segmenter and Stanford Chinese segmenters were used as the baseline methods.

4.1.3 Parameter settings

The parameters are tuned on held-out data sets. The maximum length of foreign language words is set to 4. For the PYP model, the base distribution adopts the formula in (Chung and Gildea, 2009), and the strength parameter is set to 1.0, and the discount is set to 1.0×10^{-6} .

For bilingual segmentation, the size of the alignment window is set to 6; the probability λ_ϕ of foreign language words being generated by an empty

²<http://www.itl.nist.gov/iad/mig/tests/mt/2006/>

³It also contains a small number of web blogs

Method	Accuracy		Time	
	CHLD.	MSR	CHLD.	MSR
NPY(bigram) ^a	0.750	0.802	17 m	–
NPY(trigram) ^a	0.757	0.807	–	–
HDP(bigram) ^b	0.723	–	10 h	–
Fitness ^c	–	0.667	–	–
Prop.(unigram)	0.729	0.804	3 s	50 s
Prop.(bigram)	0.774	0.806	15 s	2530 s

^a by (Mochihashi et al.,2009);

^b by (Goldwater et al.,2009);

^c by (Zhao and Kit, 2008).

Table 3: Results on Monolingual Corpora.

English word, was set to 0.3.

The training was started from assuming that there was no previous segmentations on each sentence (pair), and the number of iterations was fixed. It was set to 3 for the monolingual unigram model, and 2 for the bilingual unigram model, which provided slightly higher BLEU scores on the development set than the other settings. The monolingual bigram model, however, was slower to converge, so we started it from the segmentations of the unigram model, and using 10 iterations.

4.2 Monolingual Segmentation Results

In monolingual segmentation, the proposed methods with both unigram and bigram models were tested. Experimental results show that they are competitive to state-of-the-art baselines in both accuracy and speed (Table 3). Note that the comparison of speed is only for reference because the times are obtained from their respective papers.

4.3 Bilingual Segmentation Results

Table 4 presents the BLEU scores for Moses using different segmentation methods. Each experiment was performed three times. The proposed method with monolingual bigram model performed poorly on the Chinese monolingual segmentation task; thus, it was not tested. We intended to test (Mochihashi et al., 2009), but found it impracticable on large-scale corpora.

The experimental results show that the proposed UWS methods are comparable to the Stanford segmenters on the OpenMT06 corpus, while achieves a 0.96 BLEU increase on the PatentMT9 corpus. This is because this corpus is out-of-domain for the supervised segmenters. The CTB and PKU Stanford segmenter were both trained on annotated news text, which was the major domain of OpenMT06.

Method	BLEU	
	OpenMT06	PatentMT9
Character	29.50 ± 0.03	28.36 ± 0.09
LDC	31.33 ± 0.10	30.22 ± 0.14
Stanford(CTB)	31.68 ± 0.25	30.77 ± 0.13
Stanford(PKU)	31.54 ± 0.13	30.86 ± 0.04
Prop.(mono.)	31.47 ± 0.18	31.62 ± 0.06
Prop.(biling.)	31.61 ± 0.14	31.73 ± 0.05

Table 4: Results on Bilingual Corpora.

Method	Time	
	OpenMT06	PatentMT9
Prop.(mono.)	28 m	1 h 01 m
Prop.(biling.)	2 h 25 m	5 h 02 m

Table 5: Time Costs on Bilingual Corpora.

Table 5 presents the run times of the proposed methods on the bilingual corpora. The program is single threaded and implemented in C++. The time cost of the bilingual models is about 5 times that of the monolingual model, which is consistent with the complexity analysis in Section 3.

5 Conclusion

This paper is devoted to large-scale Chinese UWS for SMT. An efficient unified monolingual and bilingual UWS method is proposed and applied to large-scale bilingual corpora.

Complexity analysis shows that our method is capable of scaling to large-scale corpora. This was verified by experiments on a corpus of 1-million sentence pairs on which traditional MCMC approaches would struggle (Xu et al., 2008).

The proposed method does not require any annotated data, but the SMT system with it can achieve comparable performance compared to state-of-the-art supervised word segmenters trained on precious annotated data. Moreover, the proposed method yields 0.96 BLEU improvement relative to supervised word segmenters on an out-of-domain corpus. Thus, we believe that the proposed method would benefit SMT related to low-resource languages where annotated data are scarce, and would also find application in domains that differ too greatly from the domains on which supervised word segmenters were trained.

In future research, we plan to improve the bilingual UWS through applying VB and integrating more accurate alignment models such as HMM models and IBM model 4.

References

- Michael R Brent. 1999. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34(1-3):71–105.
- Peter F Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational linguistics*, 19(2):263–311.
- Pi-Chuan Chang, Michel Galley, and Christopher D Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *Proceedings of the 3rd Workshop on Statistical Machine Translation*, pages 224–232. Association for Computational Linguistics.
- Tagyoung Chung and Daniel Gildea. 2009. Unsupervised tokenization for machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 718–726. Association for Computational Linguistics.
- Thomas Emerson. 2005. The second international chinese word segmentation bakeoff. In *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing*, volume 133.
- Sharon Goldwater, Thomas L Griffiths, and Mark Johnson. 2006. Contextual dependencies in unsupervised word segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 673–680. Association for Computational Linguistics.
- 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.
- Isao Goto, Bin Lu, Ka Po Chow, Eiichiro Sumita, and Benjamin K Tsou. 2011. Overview of the patent machine translation task at the NTCIR-9 workshop. In *Proceedings of NTCIR*, volume 9, pages 559–578.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180. Association for Computational Linguistics.
- Brian MacWhinney and Catherine Snow. 1985. The child language data exchange system. *Journal of child language*, 12(2):271–296.
- Daichi Mochihashi, Takeshi Yamada, and Naonori Ueda. 2009. Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. 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: Volume 1-Volume 1*, pages 100–108. Association for Computational Linguistics.
- Graham Neubig, Masato Mimura, Shinsuke Mori, and Tatsuya Kawahara. 2010. Learning a language model from continuous speech. In *InterSpeech*, pages 1053–1056.
- ThuyLinh Nguyen, Stephan Vogel, and Noah A Smith. 2010. Nonparametric word segmentation for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 815–823. Association for Computational Linguistics.
- Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. 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.
- Michael Paul. 2008. Overview of the IWSLT 2008 evaluation campaign. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 1–17.
- Lawrence R Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.
- Jenny R Saffran, Richard N Aslin, and Elissa L Newport. 1996. Statistical learning by 8-month-old infants. *Science*, 274(5294):1926–1928.
- Yee Whye Teh and Michael I Jordan. 2010. Hierarchical Bayesian nonparametric models with applications. *Bayesian Nonparametrics: Principles and Practice*, pages 158–207.
- Yee Whye Teh. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting on Association for Computational Linguistics*, pages 985–992. Association for Computational Linguistics.
- Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter for SIGHAN Bakeoff 2005. In *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing*, volume 171. Jeju Island, Korea.

Anand Venkataraman. 2001. A statistical model for word discovery in transcribed speech. *Computational Linguistics*, 27(3):351–372.

Jia Xu, Jianfeng Gao, Kristina Toutanova, and Hermann Ney. 2008. Bayesian semi-supervised Chinese word segmentation for statistical machine translation. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 1017–1024. Association for Computational Linguistics.

Hai Zhao and Chunyu Kit. 2008. An empirical comparison of goodness measures for unsupervised chinese word segmentation with a unified framework. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing*, pages 9–16.

EM Decipherment for Large Vocabularies

Malte Nuhn and Hermann Ney

Human Language Technology and Pattern Recognition
Computer Science Department, RWTH Aachen University, Aachen, Germany
<surname>@cs.rwth-aachen.de

Abstract

This paper addresses the problem of EM-based decipherment for large vocabularies. Here, decipherment is essentially a tagging problem: Every cipher token is tagged with some plaintext type. As with other tagging problems, this one can be treated as a Hidden Markov Model (HMM), only here, the vocabularies are large, so the usual $\mathcal{O}(NV^2)$ exact EM approach is infeasible. When faced with this situation, many people turn to sampling. However, we propose to use a type of approximate EM and show that it works well. The basic idea is to collect fractional counts only over a small subset of links in the forward-backward lattice. The subset is different for each iteration of EM. One option is to use beam search to do the subsetting. The second method restricts the successor words that are looked at, for each hypothesis. It does this by consulting pre-computed tables of likely n -grams and likely substitutions.

1 Introduction

The decipherment of probabilistic substitution ciphers (ciphers in which each plaintext token can be substituted by any cipher token, following a distribution $p(f|e)$, cf. Table 2) can be seen as an important step towards decipherment for MT. This problem has not been studied explicitly before. Scaling to larger vocabularies for probabilistic substitution ciphers decipherment is a difficult problem: The algorithms for 1:1 or homophonic substitution ciphers are not applicable, and standard algorithms like EM training become intractable when vocabulary sizes go beyond a few hundred words. In this paper we present an effi-

cient EM based training procedure for probabilistic substitution ciphers which provides high decipherment accuracies while having low computational requirements. The proposed approach allows using high order n -gram language models, and is scalable to large vocabulary sizes. We show improvements in decipherment accuracy in a variety of experiments (including MT) while being computationally more efficient than previous published work on EM-based decipherment.

2 Related Work

Several methods exist for deciphering 1:1 substitution ciphers: Ravi and Knight (2008) solve 1:1 substitution ciphers by formulating the decipherment problem as an integer linear program. Corlett and Penn (2010) solve the same problem using A^* search. Nuhn et al. (2013) present a beam search approach that scales to large vocabulary and high order language models. Even though being successful, these algorithms are not applicable to probabilistic substitution ciphers, or any of its extensions as they occur in decipherment for machine translation.

EM training for probabilistic ciphers was first covered in Ravi and Knight (2011). Nuhn et al. (2012) have given an approximation to exact EM training using context vectors, allowing to training models even for larger vocabulary sizes. Ravi (2013) report results on the OPUS subtitle corpus using an elaborate hash sampling technique, based on n -gram language models and context vectors, that is computationally very efficient.

Conventional beam search is a well studied topic: Huang et al. (1992) present beam search for automatic speech recognition, using fine-grained pruning procedures. Similarly, Young and Young (1994) present an HMM toolkit, including pruned forward-backward EM training. Pal et al. (2006) use beam search for training of CRFs.

Method	Publications	Complexity
EM Full	(Knight et al., 2006), (Ravi and Knight, 2011)	$\mathcal{O}(NV^n)$
EM Fixed Candidates	(Nuhn et al., 2012)	$\mathcal{O}(N)$
EM Beam	This Work	$\mathcal{O}(NV)$
EM Lookahead	This Work	$\mathcal{O}(N)$

Table 1: Different approximations to exact EM training for decipherment. N is the cipher sequence length, V the size of the target vocabulary, and n the order of the language model.

The main contribution of this work is the pre-selection beam search that—to the best of our knowledge—was not known in literature before, and serves as an important step to applying EM training to the large vocabulary decipherment problem. Table 1 gives an overview of the EM based methods. More details are given in Section 3.2.

3 Probabilistic Substitution Ciphers

We define probabilistic substitutions ciphers using the following generative story for ciphertext sequences f_1^N :

1. Stochastically generate a plaintext sequence e_1^N according to a bigram¹ language model.
2. For each plaintext token e_n choose a substitution f_n with probability $P(f_n|e_n, \vartheta)$.

This generative story corresponds to the model

$$p(e_1^N, f_1^N, \vartheta) = p(e_1^N) \cdot p(f_1^N|e_1^N, \vartheta), \quad (1)$$

with the zero-order membership model

$$p(f_1^N|e_1^N, \vartheta) = \prod_{n=1}^N p_{lex}(f_n|e_n, \vartheta) \quad (2)$$

with parameters $p(f|e, \vartheta) \equiv \vartheta_{f|e}$ and normalization constraints $\forall e \sum_f \vartheta_{f|e} = 1$, and first-order plaintext sequence model

$$P(e_1^N) = \prod_{n=1}^N p_{LM}(e_n|e_{n-1}). \quad (3)$$

Thus, the probabilistic substitution cipher can be seen as a Hidden Markov Model. Table 2 gives an overview over the model. We want to find those parameters ϑ that maximize the marginal distribution $p(f_1^N|\vartheta)$:

$$\vartheta = \arg \max_{\vartheta'} \left\{ \sum_{[e_1^N]} p(f_1^N, e_1^N|\vartheta') \right\} \quad (4)$$

¹This can be generalized to n -gram language models.

After we obtained the parameters ϑ we can obtain e_1^N as the Viterbi decoding $\arg \max_{e_1^N} \{p(e_1^N|f_1^N, \vartheta)\}$.

3.1 Exact EM training

In the decipherment setting, we are given the observed ciphertext f_1^N and the model $p(f_1^N|e_1^N, \vartheta)$ that explains how the observed ciphertext has been generated given a latent plaintext e_1^N . Marginalizing the unknown e_1^N , we would like to obtain the maximum likelihood estimate of ϑ as specified in Equation 4. We iteratively compute the maximum likelihood estimate by applying the EM algorithm (Dempster et al., 1977):

$$\tilde{\vartheta}_{f|e} = \frac{\sum_{n:f_n=f} p_n(e|f_1^N, \vartheta)}{\sum_f \sum_{n:f_n=f} p_n(e|f_1^N, \vartheta)} \quad (5)$$

with

$$p_n(e|f_1^N, \vartheta) = \sum_{[e_1^N:e_n=e]} p(e_1^N|f_1^N, \vartheta) \quad (6)$$

being the posterior probability of observing the plaintext symbol e at position n given the ciphertext sequence f_1^N and the current parameters ϑ . $p_n(e|f_1^N, \vartheta)$ can be efficiently computed using the forward-backward algorithm.

3.2 Approximations to EM-Training

The computational complexity of EM training stems from the sum $\sum_{[e_1^N:e_n=e]}$ contained in the posterior $p_n(e|f_1^N, \vartheta)$. However, we can approximate this sum (and hope that the EM training procedure is still working) by only evaluating the dominating terms, i.e. we only evaluate the sum for sequences e_1^N that have the *largest* contributions to $\sum_{[e_1^N:e_n=e]}$. Note that due to this approximation, the new parameter estimates in Equation 5 can become zero. This is a critical issue, since pairs (e, f) with $p(f|e) = 0$ cannot recover from

Sequence of cipher tokens	:	f_1^N	=	f_1, \dots, f_N
Sequence of plaintext tokens	:	e_1^N	=	e_1, \dots, e_N
Joint probability	:	$p(f_1^N, e_1^N \vartheta)$	=	$p(e_1^N) \cdot p(f_1^N e_1^N, \vartheta)$
Language model	:	$p(e_1^N)$	=	$\prod_{n=1}^N p_{LM}(e_n e_{n-1})$
Membership probabilities	:	$p(f_1^N e_1^N, \vartheta)$	=	$\prod_{n=1}^N p_{lex}(f_n e_n, \vartheta)$
Parameter Set	:			$\vartheta = \{\vartheta_{f e}\}, p(f e, \vartheta) = \vartheta_{f e}$
Normalization	:			$\forall e : \sum_f \vartheta_{f e} = 1$
Probability of cipher sequence	:	$p(f_1^N \vartheta)$	=	$\sum_{[e_1^N]} p(f_1^N, e_1^N \vartheta)$

Table 2: Definition of the probabilistic substitution cipher model. In contrast to simple or homophonic substitution ciphers, each plaintext token can be substituted by multiple cipher text tokens. The parameter $\vartheta_{f|e}$ represents the probability of substituting token e with token f .

acquiring zero probability in some early iteration. In order to allow the lexicon to recover from these zeros, we use a smoothed lexicon $p_{lex}(f|e) = \lambda p_{lex}(f|e) + (1 - \lambda)/|V_f|$ with $\lambda = 0.9$ when conducting the *E*-Step.

3.2.1 Beam Search

Instead of evaluating the sum for terms with the *exact* largest contributions, we restrict ourselves to terms that are *likely* to have a large contribution to the sum, dropping any guarantees about the actual contribution of these terms.

Beam search is a well known algorithm related to this idea: We build up sequences e_1^c with growing cardinality c . For each cardinality, only a set of the B most promising hypotheses is kept. Then for each active hypothesis of cardinality c , all possible extensions with substitutions $f_{c+1} \rightarrow e_{c+1}$ are explored. Then in turn only the best B out of the resulting $B \cdot V_e$ many hypotheses are kept and the algorithm continues with the next cardinality. Reaching the full cardinality N , the algorithm explored $B \cdot N \cdot V_e$ many hypotheses, resulting in a complexity of $\mathcal{O}(BNV_e)$.

Even though EM training using beam search works well, it still suffers from exploring *all* V_e possible extensions for each active hypothesis, and thus scaling linearly with the vocabulary size. Due to that, standard beam search EM training is too slow to be used in the decipherment setting.

3.2.2 Preselection Search

Instead of evaluating all substitutions $f_{c+1} \rightarrow e_{c+1} \in V_e$, this algorithm only expands a fixed number of candidates: For a hypothesis ending in

a language model state σ , we only look at B_{LM} many successor words e_{c+1} with the highest LM probability $p_{LM}(e_{c+1} | \sigma)$ and at B_{lex} many successor words e_{c+1} with the highest lexical probability $p_{lex}(f_{c+1} | e_{c+1})$. Altogether, for each hypothesis we only look at $(B_{LM} + B_{lex})$ many successor states. Then, just like in the standard beam search approach, we prune all explored new hypotheses and continue with the pruned set of B many hypotheses. Thus, for a cipher of length N we only explore $N \cdot B \cdot (B_{LM} + B_{lex})$ many hypotheses.²

Intuitively speaking, our approach solves the EM training problem for decipherment using large vocabularies by focusing only on those substitutions that either seem likely due to the language model (“What word is likely to follow the current partial decipherment?”) or due to the lexicon model (“Based on my knowledge about the current cipher token, what is the most likely substitution?”).

In order to efficiently find the maximizing e for $p_{LM}(e | \sigma)$ and $p_{lex}(f | e)$, we build a lookup table that contains for each language model state σ the B_{LM} best successor words e , and a separate lookup table that contains for each source word f the B_{lex} highest scoring tokens e . The language model lookup table remains constant during all iterations, while the lexicon lookup table needs to be updated between each iteration.

Note that the size of the LM lookup table scales linearly with the number of language model states. Thus the memory requirements for the lookup ta-

²We always use $B = 100$, $B_{lex} = 5$, and $B_{LM} = 50$.

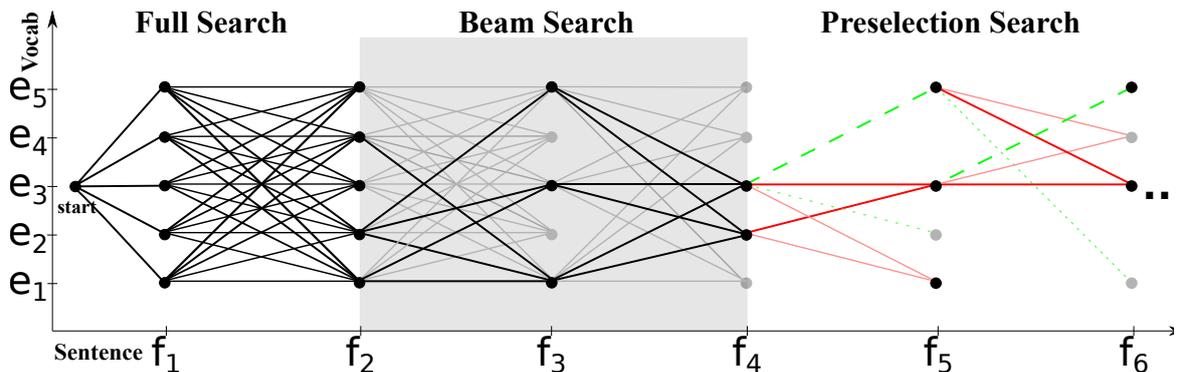


Figure 1: Illustration of the search space explored by full search, beam search, and preselection search. Full search keeps all possible hypotheses at cardinality c and explores all possible substitutions at $(c + 1)$. Beam search only keeps the B most promising hypotheses and then selects the best new hypotheses for cardinality $(c + 1)$ from *all* possible substitutions. Preselection search keeps only the B best hypotheses for every cardinality c and only looks at the $(B_{lex} + B_{LM})$ most promising substitutions for cardinality $(c + 1)$ based on the current lexicon (B_{lex} dashed lines) and language model (B_{LM} solid lines).

Name	Lang.	Sent.	Words	Voc.
VERBMOBIL	English	27,862	294,902	3,723
OPUS	Spanish	13,181	39,185	562
	English	19,770	61,835	411

Table 3: Statistics of the corpora used in this paper: The VERBMOBIL corpus is used to conduct experiments on simple substitution ciphers, while the OPUS corpus is used in our Machine Translation experiments.

ble do not form a practical problem of our approach. Figure 1 illustrates full search, beam search, and our proposed method.

4 Experimental Evaluation

We first show experiments for data in which the underlying model is an actual 1:1 substitution cipher. In this case, we report the word accuracy of the final decipherment. We then show experiments for a simple machine translation task. Here we report translation quality in BLEU. The corpora used in this paper are shown in Table 3.

4.1 Simple Substitution Ciphers

In this set of experiments, we compare the exact EM training to the approximations presented in this paper. We use the English side of the German-English VERBMOBIL corpus (Wahlster, 2000) to construct a word substitution cipher, by substituting every word type with a unique number. In order to have a non-parallel setup, we train language

Vocab	LM	Method	Acc.[%]	Time[h]
200	2	exact	97.19	224.88
		beam	98.87	9.04
		presel.	98.50	4.14
500	2	beam	92.12	24.27
		presel.	92.16	4.70
3 661	3	beam	91.16	302.81
		presel.	90.92	19.68
	4	presel.	92.14	23.72

Table 4: Results for simple substitution ciphers based on the VERBMOBIL corpus using exact, beam, and preselection EM. Exact EM is not tractable for vocabulary sizes above 200.

models of order 2, 3 and 4 on the first half of the corpus and use the second half as ciphertext. Table 4 shows the results of our experiments.

Since exact EM is not tractable for vocabulary sizes beyond 200 words, we train word classes on the whole corpus and map the words to classes (consistent along the first and second half of the corpus). By doing this, we create new simple substitution ciphers with smaller vocabularies of size 200 and 500. For the smallest setup, we can directly compare all three EM variants. We also include experiments on the original corpus with vocabulary size of 3661. When comparing exact EM training with beam- and preselection EM training, the first thing we notice is that it takes about 20 times longer to run the exact EM training than training with beam EM, and about 50 times longer than the preselection EM training. Interestingly,

Model	Method	BLEU [%]	Runtime
2-gram	Exact EM(Ravi and Knight, 2011)	15.3	850.0h
whole segment lm	Exact EM(Ravi and Knight, 2011)	19.3	850.0h
2-gram	Preselection EM (This work)	15.7	1.8h
3-gram	Preselection EM (This work)	19.5	1.9h

Table 5: Comparison of MT performance (BLEU scores) and efficiency (running time in CPU hours) on the Spanish/English OPUS corpus using only non-parallel corpora for training.

the accuracy of the approximations to exact EM training is better than that of the exact EM training. Even though this needs further investigation, it is clear that the pruned versions of EM training find sparser distributions $p_{lex}(f|e)$: This is desirable in this set of experiments, and could be the reason for improved performance.

For larger vocabularies, exact EM training is not tractable anymore. We thus constrain ourselves to running experiments with beam and preselection EM training only. Here we can see that the runtime of the preselection search is roughly the same as when running on a smaller vocabulary, while the beam search runtime scales almost linearly with the vocabulary size. For the full vocabulary of 3661 words, preselection EM using a 4-gram LM needs less than 7% of the time of beam EM with a 3-gram LM and performs by 1% better in symbol accuracy.

To summarize: Beam search EM is an order of magnitude faster than exact EM training while even increasing decipherment accuracy. Our new preselection search method is in turn orders of magnitudes faster than beam search EM while even being able to outperform exact EM and beam EM by using higher order language models. We were thus able to scale the EM decipherment to larger vocabularies of several thousand words. The runtime behavior is also consistent with the computational complexity discussed in Section 3.2.

4.2 Machine Translation

We show that our algorithm is directly applicable to the decipherment problem for machine translation. We use the same simplified translation model as presented by Ravi and Knight (2011). Because this translation model allows insertions and deletions, hypotheses of different cardinalities coexist during search. We extend our search approach such that pruning is done for each cardinality sep-

arately. Other than that, we use the same preselection search procedure as used for the simple substitution cipher task.

We run experiments on the opus corpus as presented in (Tiedemann, 2009). Table 5 shows previously published results using EM together with the results of our new method:

(Ravi and Knight, 2011) is the only publication that reports results using exact EM training and *only* n -gram language models on the target side: It has an estimated runtime of 850h. All other published results (using EM training and Bayesian inference) use context vectors as an additional source of information: This might be an explanation why Nuhn et al. (2012) and Ravi (2013) are able to outperform exact EM training as reported by Ravi and Knight (2011). (Ravi, 2013) reports the most efficient method so far: It only consumes about 3h of computation time. However, as mentioned before, those results are not directly comparable to our work, since they use additional context information on the target side.

Our algorithm clearly outperforms the exact EM training in run time, and even slightly improves performance in BLEU. Similar to the simple substitution case, the improved performance might be caused by inferring a sparser distribution $p_{lex}(f|e)$. However, this requires further investigation.

5 Conclusion

We have shown a conceptually consistent and easy to implement EM based training method for decipherment that outperforms exact and beam search EM training for simple substitution ciphers and decipherment for machine translation, while reducing training time to a fraction of exact and beam EM. We also point out that the preselection method presented in this paper is not restricted to word based translation models and can also be applied to phrase based translation models.

References

- Eric Corlett and Gerald Penn. 2010. An exact A* method for deciphering letter-substitution ciphers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1040–1047, Uppsala, Sweden, July. The Association for Computational Linguistics.
- Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B*, 39.
- Xuedong Huang, Fileno Alleva, Hsiao wuen Hon, Meiyuh Hwang, and Ronald Rosenfeld. 1992. The sphinx-ii speech recognition system: An overview. *Computer, Speech and Language*, 7:137–148.
- Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. 2006. Unsupervised Analysis for Decipherment Problems. In *Proceedings of the COLING/ACL on Main conference poster sessions*, COLING-ACL '06, pages 499–506, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Malte Nuhn, Arne Mauser, and Hermann Ney. 2012. Deciphering foreign language by combining language models and context vectors. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 156–164, Jeju, Republic of Korea, July. Association for Computational Linguistics.
- Malte Nuhn, Julian Schamper, and Hermann Ney. 2013. Beam search for solving substitution ciphers. In *Annual Meeting of the Assoc. for Computational Linguistics*, pages 1569–1576, Sofia, Bulgaria, August.
- Chris Pal, Charles Sutton, and Andrew McCallum. 2006. Sparse forward-backward using minimum divergence beams for fast training of conditional random fields. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.
- Sujith Ravi and Kevin Knight. 2008. Attacking decipherment problems optimally with low-order n-gram models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 812–819, Honolulu, Hawaii. Association for Computational Linguistics.
- Sujith Ravi and Kevin Knight. 2011. Deciphering foreign language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 12–21, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Sujith Ravi. 2013. Scalable decipherment for machine translation via hash sampling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 362–371, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Jörg Tiedemann. 2009. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria.
- Wolfgang Wahlster, editor. 2000. *Verbmobil: Foundations of speech-to-speech translations*. Springer-Verlag, Berlin.
- S.J. Young and S.J. Young. 1994. The htk hidden markov model toolkit: Design and philosophy. *Entropy Cambridge Research Laboratory, Ltd*, 2:2–44.

XMEANT: Better semantic MT evaluation without reference translations

Lo, Chi-kiu Beloucif, Meriem Saers, Markus Wu, Dekai

HKUST

Human Language Technology Center

Department of Computer Science and Engineering

Hong Kong University of Science and Technology

{jackielo|mbeloucif|masaers|dekai}@cs.ust.hk

Abstract

We introduce XMEANT—a new *cross-lingual* version of the semantic frame based MT evaluation metric MEANT—which can correlate even more closely with human adequacy judgments than monolingual MEANT and eliminates the need for expensive human references. Previous work established that MEANT reflects translation adequacy with state-of-the-art accuracy, and optimizing MT systems against MEANT robustly improves translation quality. However, to go beyond tuning weights in the loglinear SMT model, a cross-lingual objective function that can deeply integrate semantic frame criteria into the MT training pipeline is needed. We show that cross-lingual XMEANT outperforms monolingual MEANT by (1) replacing the monolingual context vector model in MEANT with simple translation probabilities, and (2) incorporating bracketing ITG constraints.

1 Introduction

We show that XMEANT, a new cross-lingual version of MEANT (Lo et al., 2012), correlates with human judgment even more closely than MEANT for evaluating MT adequacy via semantic frames, despite discarding the need for expensive human reference translations. XMEANT is obtained by (1) using simple lexical translation probabilities, instead of the monolingual context vector model used in MEANT for computing the semantic role fillers similarities, and (2) incorporating bracketing ITG constraints for word alignment within the semantic role fillers. We conjecture that the reason that XMEANT correlates more closely with human adequacy judgement than MEANT is that on the one hand, the semantic structure of the MT output is closer to that of the input sentence

than that of the reference translation, and on the other hand, the BITG constraints the word alignment more accurately than the heuristic bag-of-word aggregation used in MEANT. Our results suggest that MT translation adequacy is more accurately evaluated via the cross-lingual semantic frame similarities of the input and the MT output which may obviate the need for expensive human reference translations.

The MEANT family of metrics (Lo and Wu, 2011a, 2012; Lo et al., 2012) adopt the principle that a good translation is one where a human can successfully understand the central meaning of the foreign sentence as captured by the basic event structure: “*who did what to whom, when, where and why*” (Pradhan et al., 2004). MEANT measures similarity between the MT output and the reference translations by comparing the similarities between the semantic frame structures of output and reference translations. It is well established that the MEANT family of metrics correlates better with human adequacy judgments than commonly used MT evaluation metrics (Lo and Wu, 2011a, 2012; Lo et al., 2012; Lo and Wu, 2013b; Macháček and Bojar, 2013). In addition, the translation adequacy across different genres (ranging from formal news to informal web forum and public speech) and different languages (English and Chinese) is improved by replacing BLEU or TER with MEANT during parameter tuning (Lo et al., 2013a; Lo and Wu, 2013a; Lo et al., 2013b).

In order to continue driving MT towards better translation adequacy by deeply integrating semantic frame criteria into the MT training pipeline, it is necessary to have a *cross-lingual semantic objective function* that assesses the semantic frame similarities of input and output sentences. We therefore propose XMEANT, a cross-lingual MT evaluation metric, that modifies MEANT using (1) simple translation probabilities (in our experiments,

from quick IBM-1 training), to replace the monolingual context vector model in MEANT, and (2) constraints from BITGs (bracketing ITGs). We show that XMEANT assesses MT adequacy more accurately than MEANT (as measured by correlation with human adequacy judgement) without the need for expensive human reference translations in the output language.

2 Related Work

2.1 MT evaluation metrics

Surface-form oriented metrics such as BLEU (Papineni et al., 2002), NIST (Doddington, 2002), METEOR (Banerjee and Lavie, 2005), CDER (Leusch et al., 2006), WER (Nießen et al., 2000), and TER (Snover et al., 2006) do not correctly reflect the meaning similarities of the input sentence. In fact, a number of large scale meta-evaluations (Callison-Burch et al., 2006; Koehn and Monz, 2006) report cases where BLEU strongly disagrees with human judgments of translation adequacy.

This has caused a recent surge of work to develop better ways to automatically measure MT adequacy. Owczarzak et al. (2007a,b) improved correlation with human *fluency* judgments by using LFG to extend the approach of evaluating syntactic dependency structure similarity proposed by Liu and Gildea (2005), but did not achieve higher correlation with human *adequacy* judgments than metrics like METEOR. TINE (Rios et al., 2011) is a recall-oriented metric which aims to preserve the basic event structure but it performs comparably to BLEU and worse than METEOR on correlation with human adequacy judgments. ULC (Giménez and Márquez, 2007, 2008) incorporates several semantic features and shows improved correlation with human judgement on translation quality (Callison-Burch et al., 2007, 2008) but no work has been done towards tuning an SMT system using a pure form of ULC perhaps due to its expensive run time. Similarly, SPEDE (Wang and Manning, 2012) predicts the edit sequence for matching the MT output to the reference via an integrated probabilistic FSM and PDA model. Sagan (Castillo and Estrella, 2012) is a semantic textual similarity metric based on a complex textual entailment pipeline. These aggregated metrics require sophisticated feature extraction steps, contain several dozens of parameters to tune, and employ expensive linguistic resources like WordNet

1. Apply an automatic shallow semantic parser to both the references and MT output. (Figure 2 shows examples of automatic shallow semantic parses on both reference and MT.)
2. Apply the maximum weighted bipartite matching algorithm to align the semantic frames between the references and MT output according to the lexical similarities of the predicates.
3. For each pair of the aligned frames, apply the maximum weighted bipartite matching algorithm to align the arguments between the reference and MT output according to the lexical similarity of role fillers.
4. Compute the weighted f-score over the matching role labels of these aligned predicates and role fillers according to the following definitions:

$$\begin{aligned}
 q_{i,j}^0 &\equiv \text{ARG } j \text{ of aligned frame } i \text{ in MT} \\
 q_{i,j}^1 &\equiv \text{ARG } j \text{ of aligned frame } i \text{ in REF} \\
 w_i^0 &\equiv \frac{\text{\#tokens filled in aligned frame } i \text{ of MT}}{\text{total \#tokens in MT}} \\
 w_i^1 &\equiv \frac{\text{\#tokens filled in aligned frame } i \text{ of REF}}{\text{total \#tokens in REF}} \\
 w_{\text{pred}} &\equiv \text{weight of similarity of predicates} \\
 w_j &\equiv \text{weight of similarity of ARG } j \\
 s_{i,\text{pred}} &\equiv \text{predicate similarity in aligned frame } i \\
 s_{i,j} &\equiv \text{ARG } j \text{ similarity in aligned frame } i \\
 \text{precision} &= \frac{\sum_i w_i^0 \frac{w_{\text{pred}} s_{i,\text{pred}} + \sum_j w_j s_{i,j}}{w_{\text{pred}} + \sum_j w_j |q_{i,j}^0|}}{\sum_i w_i^0} \\
 \text{recall} &= \frac{\sum_i w_i^1 \frac{w_{\text{pred}} s_{i,\text{pred}} + \sum_j w_j s_{i,j}}{w_{\text{pred}} + \sum_j w_j |q_{i,j}^1|}}{\sum_i w_i^1} \\
 \text{MEANT} &= \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}
 \end{aligned}$$

Figure 1: Monolingual MEANT algorithm.

or paraphrase tables; the expensive training, tuning, and/or running time makes them hard to incorporate into the MT development cycle.

2.2 The MEANT family of metrics

MEANT (Lo et al., 2012), which is the weighted f-score over the matched semantic role labels of the automatically aligned semantic frames and role fillers, that outperforms BLEU, NIST, METEOR, WER, CDER and TER in correlation with human adequacy judgments. MEANT is easily portable to other languages, requiring only an automatic semantic parser and a large monolingual corpus in the output language for identifying the semantic structures and the lexical similarity between the semantic role fillers of the reference and translation.

Figure 1 shows the algorithm and equations for computing MEANT. $q_{i,j}^0$ and $q_{i,j}^1$ are the argument of type j in frame i in MT and REF respectively. w_i^0 and w_i^1 are the weights for frame i in MT/REF respectively. These weights estimate the degree of contribution of each frame to the overall meaning of the sentence. w_{pred} and w_j are the weights of the lexical similarities of the predicates and role fillers of the arguments of type j of all frame between the reference translations and the MT out-

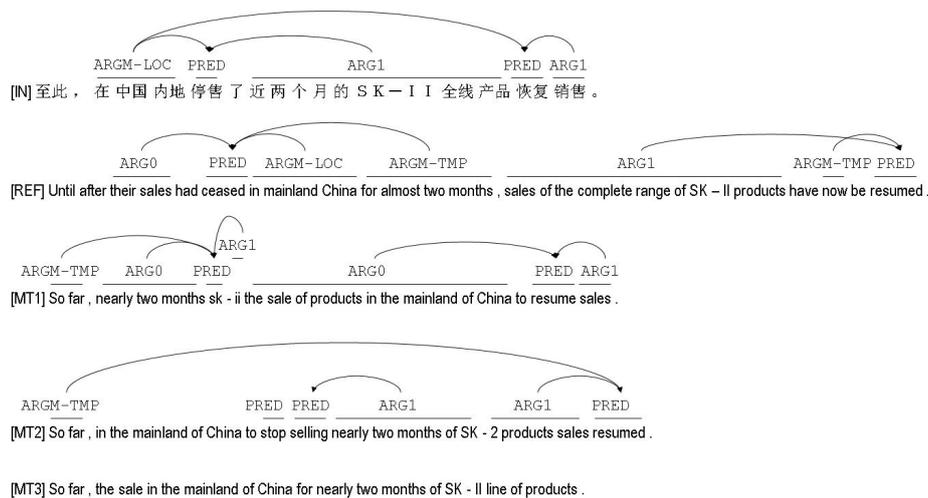


Figure 2: Examples of automatic shallow semantic parses. The input is parsed by a Chinese automatic shallow semantic parser. The reference and MT output are parsed by an English automatic shallow semantic parser. There are no semantic frames for MT3 since the system decided to drop the predicate.

put. There is a total of 12 weights for the set of semantic role labels in MEANT as defined in Lo and Wu (2011b). For MEANT, they are determined using supervised estimation via a simple grid search to optimize the correlation with human adequacy judgments (Lo and Wu, 2011a). For UMEANT (Lo and Wu, 2012), they are estimated in an unsupervised manner using relative frequency of each semantic role label in the references and thus UMEANT is useful when human judgments on adequacy of the development set are unavailable.

$s_{i,\text{pred}}$ and $s_{i,j}$ are the lexical similarities based on a context vector model of the predicates and role fillers of the arguments of type j between the reference translations and the MT output. Lo et al. (2012) and Tumuluru et al. (2012) described how the lexical and phrasal similarities of the semantic role fillers are computed. A subsequent variant of the aggregation function inspired by Mihalcea et al. (2006) that normalizes phrasal similarities according to the phrase length more accurately was used in more recent work (Lo et al., 2013a; Lo and Wu, 2013a; Lo et al., 2013b). In this paper, we employ a newer version of MEANT that uses f-score to aggregate individual token similarities into the composite phrasal similarities of semantic role fillers, as our experiments indicate this is more accurate than the previously used aggregation functions.

Recent studies (Lo et al., 2013a; Lo and Wu, 2013a; Lo et al., 2013b) show that tuning MT sys-

tems against MEANT produces more robustly adequate translations than the common practice of tuning against BLEU or TER across different data genres, such as formal newswire text, informal web forum text and informal public speech.

2.3 MT quality estimation

Evaluating cross-lingual MT quality is similar to the work of MT quality estimation (QE). Broadly speaking, there are two different approaches to QE: surface-based and feature-based.

Token-based QE models, such as those in Gandrabur et al. (2006) and Ueffing and Ney (2005) fail to assess the overall MT quality because translation goodness is not a compositional property. In contrast, Blatz et al. (2004) introduced a sentence-level QE system where an arbitrary threshold is used to classify the MT output as *good* or *bad*. The fundamental problem of this approach is that it defines QE as a binary classification task rather than attempting to measure the degree of goodness of the MT output. To address this problem, Quirk (2004) related the sentence-level correctness of the QE model to human judgment and achieved a high correlation with human judgement for a small annotated corpus; however, the proposed model does not scale well to larger data sets.

Feature-based QE models (Xiong et al., 2010; He et al., 2011; Ma et al., 2011; Specia, 2011; Avramidis, 2012; Mehdad et al., 2012; Almaghout and Specia, 2013; Avramidis and Popović, 2013; Shah et al., 2013) throw a wide range of linguistic and non-linguistic features into machine learn-

1. Apply an input language automatic shallow semantic parser to the foreign input and an output language automatic shallow semantic parser to the MT output. (Figure 2 shows examples of automatic shallow semantic parses on both foreign input and MT output. The Chinese semantic parser used in our experiments is C-ASSERT in (Fung *et al.*, 2004, 2007).)
2. Apply the maximum weighted bipartite matching algorithm to align the semantic frames between the foreign input and MT output according to the lexical translation probabilities of the predicates.
3. For each pair of the aligned frames, apply the maximum weighted bipartite matching algorithm to align the arguments between the foreign input and MT output according to the aggregated phrasal translation probabilities of the role fillers.
4. Compute the weighted f-score over the matching role labels of these aligned predicates and role fillers according to the definitions similar to those in section 2.2 except for replacing REF with IN in $q_{i,j}^1$ and w_i^1 .

Figure 3: Cross-lingual XMEANT algorithm.

ing algorithms for predicting MT quality. Although the feature-based QE system of Avramidis and Popović (2013) slightly outperformed ME-TEOR on correlation with human adequacy judgment, these “black box” approaches typically lack representational transparency, require expensive running time, and/or must be discriminatively re-trained for each language and text type.

3 XMEANT: a cross-lingual MEANT

Like MEANT, XMEANT aims to evaluate how well MT preserves the core semantics, while maintaining full representational transparency. But whereas MEANT measures lexical similarity using a monolingual context vector model, XMEANT instead substitutes simple cross-lingual lexical translation probabilities.

XMEANT differs only minimally from MEANT, as underlined in figure 3. The same weights obtained by optimizing MEANT against human adequacy judgement were used for XMEANT. The weights can also be estimated in unsupervised fashion using the relative frequency of each semantic role label in the foreign input, as in UMEANT.

To aggregate individual lexical translation probabilities into phrasal similarities between cross-lingual semantic role fillers, we compared two natural approaches to generalizing MEANT’s method of comparing semantic parses, as described below.

3.1 Applying MEANT’s f-score within semantic role fillers

The first natural approach is to extend MEANT’s f-score based method of aggregating semantic parse accuracy, so as to also apply to aggregat-

ing lexical translation probabilities *within* semantic role filler phrases. However, since we are missing structure information within the flat role filler phrases, we can no longer assume an injective mapping for aligning the tokens of the role fillers between the foreign input and the MT output. We therefore relax the assumption and thus for cross-lingual phrasal precision/recall, we align each token of the role fillers in the output/input string to the token of the role fillers in the input/output string that has the maximum lexical translation probability. The precise definition of the cross-lingual phrasal similarities is as follows:

$$\begin{aligned}
 \mathbf{e}_{i,\text{pred}} &\equiv \text{the output side of the pred of aligned frame } i \\
 \mathbf{f}_{i,\text{pred}} &\equiv \text{the input side of the pred of aligned frame } i \\
 \mathbf{e}_{i,j} &\equiv \text{the output side of the ARG } j \text{ of aligned frame } i \\
 \mathbf{f}_{i,j} &\equiv \text{the input side of the ARG } j \text{ of aligned frame } i \\
 p(e, f) &= \frac{\sqrt{t(e|f)t(f|e)}}{\sum_{e \in \mathbf{e}} \max_{f \in \mathbf{f}} p(e, f)} \\
 \text{prec}_{\mathbf{e},\mathbf{f}} &= \frac{|\mathbf{e}|}{\sum_{f \in \mathbf{f}} \max_{e \in \mathbf{e}} p(e, f)} \\
 \text{rec}_{\mathbf{e},\mathbf{f}} &= \frac{|\mathbf{f}|}{\sum_{e \in \mathbf{e}} \max_{f \in \mathbf{f}} p(e, f)} \\
 s_{i,\text{pred}} &= \frac{2 \cdot \text{prec}_{\mathbf{e}_{i,\text{pred}},\mathbf{f}_{i,\text{pred}}} \cdot \text{rec}_{\mathbf{e}_{i,\text{pred}},\mathbf{f}_{i,\text{pred}}}}{\text{prec}_{\mathbf{e}_{i,\text{pred}},\mathbf{f}_{i,\text{pred}}} + \text{rec}_{\mathbf{e}_{i,\text{pred}},\mathbf{f}_{i,\text{pred}}}} \\
 s_{i,j} &= \frac{2 \cdot \text{prec}_{\mathbf{e}_{i,j},\mathbf{f}_{i,j}} \cdot \text{rec}_{\mathbf{e}_{i,j},\mathbf{f}_{i,j}}}{\text{prec}_{\mathbf{e}_{i,j},\mathbf{f}_{i,j}} + \text{rec}_{\mathbf{e}_{i,j},\mathbf{f}_{i,j}}}
 \end{aligned}$$

where the joint probability p is defined as the harmonized the two directions of the translation table t trained using IBM model 1 (Brown *et al.*, 1993). $\text{prec}_{\mathbf{e},\mathbf{f}}$ is the precision and $\text{rec}_{\mathbf{e},\mathbf{f}}$ is the recall of the phrasal similarities of the role fillers. $s_{i,\text{pred}}$ and $s_{i,j}$ are the f-scores of the phrasal similarities of the predicates and role fillers of the arguments of type j between the input and the MT output.

3.2 Applying MEANT’s ITG bias within semantic role fillers

The second natural approach is to extend MEANT’s ITG bias on compositional reordering, so as to also apply to aggregating lexical translation probabilities *within* semantic role filler phrases. Addanki *et al.* (2012) showed empirically that cross-lingual semantic role reordering of the kind that MEANT is based upon is fully covered within ITG constraints. In Wu *et al.* (2014), we extend ITG constraints into aligning the tokens within the semantic role fillers within monolingual MEANT, thus replacing its previous monolingual phrasal aggregation heuristic. Here we borrow the

idea for the cross-lingual case, using the length-normalized inside probability at the root of a BITG biparse (Wu, 1997; Zens and Ney, 2003; Saers and Wu, 2009) as follows:

$$\begin{aligned}
G &\equiv \langle \{A\}, \mathcal{W}^0, \mathcal{W}^1, \mathcal{R}, A \rangle \\
\mathcal{R} &\equiv \{A \rightarrow [AA], A \rightarrow \langle AA \rangle, A \rightarrow e/f\} \\
p([AA]|A) &= p(\langle AA \rangle|A) = 0.25 \\
p(e/f|A) &= \frac{1}{2} \sqrt{t(e|f)t(f|e)} \\
s_{i,\text{pred}} &= \frac{1}{1 - \frac{\ln(P(A \xrightarrow{*} \mathbf{e}_{i,\text{pred}}/\mathbf{f}_{i,\text{pred}}|G))}{\max(|\mathbf{e}_{i,\text{pred}}|, |\mathbf{f}_{i,\text{pred}}|)}} \\
s_{i,j} &= \frac{1}{1 - \frac{\ln(P(A \xrightarrow{*} \mathbf{e}_{i,j}/\mathbf{f}_{i,j}|G))}{\max(|\mathbf{e}_{i,j}|, |\mathbf{f}_{i,j}|)}}
\end{aligned}$$

where G is a bracketing ITG, whose only nonterminal is A , and where \mathcal{R} is a set of transduction rules where $e \in \mathcal{W}^0 \cup \{\epsilon\}$ is an output token (or the *null* token), and $f \in \mathcal{W}^1 \cup \{\epsilon\}$ is an input token (or the *null* token). The rule probability function p is defined using fixed probabilities for the structural rules, and a translation table t trained using IBM model 1 in both directions. To calculate the inside probability of a pair of segments, $P(A \xrightarrow{*} \mathbf{e}/\mathbf{f}|G)$, we use the algorithm described in Saers et al. (2009). $s_{i,\text{pred}}$ and $s_{i,j}$ are the length normalized BITG parsing probabilities of the predicates and role fillers of the arguments of type j between the input and the MT output.

4 Results

Table 1 shows that for human adequacy judgments at the sentence level, the f-score based XMEANT (1) correlates significantly more closely than other commonly used monolingual automatic MT evaluation metrics, and (2) even correlates nearly as well as monolingual MEANT. This suggests that the semantic structure of the MT output is indeed closer to that of the input sentence than that of the reference translation.

Furthermore, the ITG-based XMEANT (1) significantly outperforms MEANT, and (2) is an automatic metric that is nearly as accurate as the HMEANT human subjective version. This indicates that BITG constraints indeed provide a more robust token alignment compared to the heuristics previously employed in MEANT. It is also consistent with results observed while estimating word alignment probabilities, where BITG constraints outperformed alignments from GIZA++ (Saers and Wu, 2009).

Table 1: Sentence-level correlation with HAJ (GALE phase 2.5 evaluation data)

<i>Metric</i>	<i>Kendall</i>
HMEANT	0.53
XMEANT (BITG)	0.51
MEANT (f-score)	0.48
XMEANT (f-score)	0.46
MEANT (2013)	0.46
NIST	0.29
BLEU/METEOR/TER/PER	0.20
CDER	0.12
WER	0.10

5 Conclusion

We have presented XMEANT, a new cross-lingual variant of MEANT, that correlates even more closely with human translation adequacy judgments than MEANT, without the expensive human references. This is (1) accomplished by replacing monolingual MEANT’s context vector model with simple translation probabilities when computing similarities of semantic role fillers, and (2) further improved by incorporating BITG constraints for aligning the tokens in semantic role fillers. While monolingual MEANT alone accurately reflects adequacy via semantic frames and optimizing SMT against MEANT improves translation, the new cross-lingual XMEANT semantic objective function moves closer toward deep integration of semantics into the MT training pipeline.

The phrasal similarity scoring has only been minimally adapted to cross-lingual semantic role fillers in this first study of XMEANT. We expect further improvements to XMEANT, but these first results already demonstrate XMEANT’s potential to drive research progress toward semantic SMT.

6 Acknowledgments

This material is based upon work supported in part by the Defense Advanced Research Projects Agency (DARPA) under BOLT contract nos. HR0011-12-C-0014 and HR0011-12-C-0016, and GALE contract nos. HR0011-06-C-0022 and HR0011-06-C-0023; by the European Union under the FP7 grant agreement no. 287658; and by the Hong Kong Research Grants Council (RGC) research grants GRF620811, GRF621008, and GRF612806. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA, the EU, or RGC.

References

- Karteek Addanki, Chi-Kiu Lo, Markus Saers, and Dekai Wu. LTG vs. ITG coverage of cross-lingual verb frame alternations. In *16th Annual Conference of the European Association for Machine Translation (EAMT-2012)*, Trento, Italy, May 2012.
- Hala Almaghout and Lucia Specia. A CCG-based quality estimation metric for statistical machine translation. In *Machine Translation Summit XIV (MT Summit 2013)*, Nice, France, 2013.
- Eleftherios Avramidis and Maja Popović. Machine learning methods for comparative and time-oriented quality estimation of machine translation output. In *8th Workshop on Statistical Machine Translation (WMT 2013)*, 2013.
- Eleftherios Avramidis. Quality estimation for machine translation output using linguistic analysis and decoding features. In *7th Workshop on Statistical Machine Translation (WMT 2012)*, 2012.
- Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, Ann Arbor, Michigan, June 2005.
- John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. Confidence estimation for machine translation. In *20th international conference on Computational Linguistics*, 2004.
- Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.
- Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of BLEU in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-2006)*, 2006.
- Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. (meta-) evaluation of machine translation. In *Second Workshop on Statistical Machine Translation (WMT-07)*, 2007.
- Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. Further meta-evaluation of machine translation. In *Third Workshop on Statistical Machine Translation (WMT-08)*, 2008.
- Julio Castillo and Paula Estrella. Semantic textual similarity for MT evaluation. In *7th Workshop on Statistical Machine Translation (WMT 2012)*, 2012.
- George Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *The second international conference on Human Language Technology Research (HLT '02)*, San Diego, California, 2002.
- Simona Gandrabur, George Foster, and Guy Lapalme. Confidence estimation for nlp applications. *ACM Transactions on Speech and Language Processing*, 2006.
- Jesús Giménez and Lluís Màrquez. Linguistic features for automatic evaluation of heterogeneous MT systems. In *Second Workshop on Statistical Machine Translation (WMT-07)*, pages 256–264, Prague, Czech Republic, June 2007.
- Jesús Giménez and Lluís Màrquez. A smorgasbord of features for automatic MT evaluation. In *Third Workshop on Statistical Machine Translation (WMT-08)*, Columbus, Ohio, June 2008.
- Yifan He, Yanjun Ma, Andy Way, and Josef van Genabith. Rich linguistic features for translation memory-inspired consistent translation. In *13th Machine Translation Summit (MT Summit XIII)*, 2011.
- Philipp Koehn and Christof Monz. Manual and automatic evaluation of machine translation between European languages. In *Workshop on Statistical Machine Translation (WMT-06)*, 2006.
- Gregor Leusch, Nicola Ueffing, and Hermann Ney. CDer: Efficient MT evaluation using block movements. In *11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-2006)*, 2006.
- Ding Liu and Daniel Gildea. Syntactic features for evaluation of machine translation. In *Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, Ann Arbor, Michigan, June 2005.
- Chi-kiu Lo and Dekai Wu. MEANT: An inexpensive, high-accuracy, semi-automatic metric for evaluating translation utility based on semantic roles. In *49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*, 2011.
- Chi-kiu Lo and Dekai Wu. SMT vs. AI redux: How semantic frames evaluate MT more accurately. In *Twenty-second International Joint Conference on Artificial Intelligence (IJCAI-11)*, 2011.
- Chi-kiu Lo and Dekai Wu. Unsupervised vs. supervised weight estimation for semantic MT evaluation metrics. In *Sixth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-6)*, 2012.
- Chi-kiu Lo and Dekai Wu. Can informal genres be better translated by tuning on automatic semantic metrics? In *14th Machine Translation Summit (MT Summit XIV)*, 2013.
- Chi-kiu Lo and Dekai Wu. MEANT at WMT 2013: A tunable, accurate yet inexpensive semantic frame based mt evaluation metric. In *8th Workshop on Statistical Machine Translation (WMT 2013)*, 2013.
- Chi-kiu Lo, Anand Karthik Tumuluru, and Dekai Wu. Fully automatic semantic MT evaluation. In *7th Workshop on Statistical Machine Translation (WMT 2012)*, 2012.
- Chi-kiu Lo, Karteek Addanki, Markus Saers, and Dekai Wu. Improving machine translation by training against an automatic semantic frame based eval-

- uation metric. In *51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, 2013.
- Chi-kiu Lo, Meriem Beloucif, and Dekai Wu. Improving machine translation into Chinese by tuning against Chinese MEANT. In *International Workshop on Spoken Language Translation (IWSLT 2013)*, 2013.
- YanJun Ma, Yifan He, Andy Way, and Josef van Genabith. Consistent translation using discriminative learning: a translation memory-inspired approach. In *49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*. Association for Computational Linguistics, 2011.
- Matouš Macháček and Ondřej Bojar. Results of the WMT13 metrics shared task. In *Eighth Workshop on Statistical Machine Translation (WMT 2013)*, Soňa, Bulgaria, August 2013.
- Yashar Mehdad, Matteo Negri, and Marcello Federico. Match without a referee: evaluating mt adequacy without reference translations. In *7th Workshop on Statistical Machine Translation (WMT 2012)*, 2012.
- Rada Mihalcea, Courtney Corley, and Carlo Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In *The Twenty-first National Conference on Artificial Intelligence (AAAI-06)*, volume 21. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.
- Sonja Nießen, Franz Josef Och, Gregor Leusch, and Hermann Ney. A evaluation tool for machine translation: Fast evaluation for MT research. In *The Second International Conference on Language Resources and Evaluation (LREC 2000)*, 2000.
- Karolina Owczarzak, Josef van Genabith, and Andy Way. Dependency-based automatic evaluation for machine translation. In *Syntax and Structure in Statistical Translation (SSST)*, 2007.
- Karolina Owczarzak, Josef van Genabith, and Andy Way. Evaluating machine translation with LFG dependencies. *Machine Translation*, 21:95–119, 2007.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *40th Annual Meeting of the Association for Computational Linguistics (ACL-02)*, pages 311–318, Philadelphia, Pennsylvania, July 2002.
- Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin, and Dan Jurafsky. Shallow semantic parsing using support vector machines. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004)*, 2004.
- Christopher Quirk. Training a sentence-level machine translation confidence measure. In *Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal, May 2004.
- Miguel Rios, Wilker Aziz, and Lucia Specia. Tine: A metric to assess MT adequacy. In *Sixth Workshop on Statistical Machine Translation (WMT 2011)*, 2011.
- Markus Saers and Dekai Wu. Improving phrase-based translation via word alignments from stochastic inversion transduction grammars. In *Third Workshop on Syntax and Structure in Statistical Translation (SSST-3)*, Boulder, Colorado, June 2009.
- Markus Saers, Joakim Nivre, and Dekai Wu. Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm. In *11th International Conference on Parsing Technologies (IWPT'09)*, Paris, France, October 2009.
- Kashif Shah, Trevor Cohn, and Lucia Specia. An investigation on the effectiveness of features for translation quality estimation. In *Machine Translation Summit XIV (MT Summit 2013)*, Nice, France, 2013.
- Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *7th Biennial Conference Association for Machine Translation in the Americas (AMTA 2006)*, pages 223–231, Cambridge, Massachusetts, August 2006.
- Lucia Specia. Exploiting objective annotations for measuring translation post-editing effort. In *15th Conference of the European Association for Machine Translation*, pages 73–80, 2011.
- Anand Karthik Tumuluru, Chi-kiu Lo, and Dekai Wu. Accuracy and robustness in measuring the lexical similarity of semantic role fillers for automatic semantic MT evaluation. In *26th Pacific Asia Conference on Language, Information, and Computation (PACLIC 26)*, 2012.
- Nicola Ueffing and Hermann Ney. Word-level confidence estimation for machine translation using phrase-based translation models. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 763–770, 2005.
- Mengqiu Wang and Christopher D. Manning. SPEDE: Probabilistic edit distance metrics for MT evaluation. In *7th Workshop on Statistical Machine Translation (WMT 2012)*, 2012.
- Dekai Wu, Chi-kiu Lo, Meriem Beloucif, and Markus Saers. IMEANT: Improving semantic frame based MT evaluation via inversion transduction grammars. Forthcoming, 2014.
- Dekai Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403, 1997.
- Deyi Xiong, Min Zhang, and Haizhou Li. Error detection for statistical machine translation using linguistic features. In *48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, 2010.
- Richard Zens and Hermann Ney. A comparative study on reordering constraints in statistical machine translation. In *41st Annual Meeting of the Association for Computational Linguistics (ACL-2003)*, pages 144–151, Stroudsburg, Pennsylvania, 2003.

Sentence Level Dialect Identification for Machine Translation System Selection

Wael Salloum, Heba Elfardy, Linda Alamir-Salloum, Nizar Habash and Mona Diab[†]
Center for Computational Learning Systems, Columbia University, New York, USA

{wael, heba, habash}@ccls.columbia.edu

[†]Department of Computer Science, The George Washington University, Washington DC, USA

†mtdiab@email.gwu.edu

Abstract

In this paper we study the use of sentence-level dialect identification in optimizing machine translation system selection when translating mixed dialect input. We test our approach on Arabic, a prototypical diglossic language; and we optimize the combination of four different machine translation systems. Our best result improves over the best single MT system baseline by 1.0% BLEU and over a strong system selection baseline by 0.6% BLEU on a blind test set.

1 Introduction

A language can be described as a set of dialects, among which one "standard variety" has a special representative status.¹ Despite being increasingly ubiquitous in informal written genres such as social media, most non-standard dialects are resource-poor compared to their standard variety. For statistical machine translation (MT), which relies on the existence of parallel data, translating from non-standard dialects is a challenge. In this paper we study the use of sentence-level dialect identification together with various linguistic features in optimizing the selection of outputs of four different MT systems on input text that includes a mix of dialects.

We test our approach on Arabic, a prototypical diglossic language (Ferguson, 1959) where the standard form of the language, Modern Standard Arabic (MSA) and the regional dialects (DA) live side-by-side and are closely related. MSA is the language used in education, scripted speech and official settings while DA is the primarily spoken

¹This paper presents work supported by the Defense Advanced Research Projects Agency (DARPA) contract No. HR0011-12-C-0014. Any opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of DARPA.

native vernacular. We consider two DAs: Egyptian and Levantine Arabic in addition to MSA. Our best system selection approach improves over our best baseline single MT system by 1.0% absolute BLEU point on a blind test set.

2 Related Work

Arabic Dialect Machine Translation. Two approaches have emerged to alleviate the problem of DA-English parallel data scarcity: using MSA as a bridge language (Sawaf, 2010; Salloum and Habash, 2011; Salloum and Habash, 2013; Sajjad et al., 2013), and using crowd sourcing to acquire parallel data (Zbib et al., 2012). Sawaf (2010) and Salloum and Habash (2013) used hybrid solutions that combine rule-based algorithms and resources such as lexicons and morphological analyzers with statistical models to map DA to MSA before using MSA-to-English MT systems. Zbib et al. (2012) obtained a 1.5M word parallel corpus of DA-English using crowd sourcing. Applied on a DA test set, a system trained on their 1.5M word corpus outperformed a system that added 150M words of MSA-English data, as well as outperforming a system with oracle DA-to-MSA pivot.

In this paper we use four MT systems that translate from DA to English in different ways. Similar to Zbib et al. (2012), we use DA-English, MSA-English and DA+MSA-English systems. Our DA-English data includes the 1.5M words created by Zbib et al. (2012). Our fourth MT system uses ELISSA, the DA-to-MSA MT tool by Salloum and Habash (2013), to produce an MSA pivot.

Dialect Identification. There has been a number of efforts on dialect identification (Biadisy et al., 2009; Zaidan and Callison-Burch, 2011; Akbacak et al., 2011; Elfardy et al., 2013; Elfardy and Diab, 2013). Elfardy et al. (2013) performed token-level dialect ID by casting the problem as a code-switching problem and treating MSA and Egyptian as two different languages. They later

used features from their token-level system to train a classifier that performs sentence-level dialect ID (Elfardy and Diab, 2013). In this paper, we use AIDA, the system of Elfardy and Diab (2013), to provide a variety of dialect ID features to train classifiers that select, for a given sentence, the MT system that produces the best translation.

System Selection and Combination in Machine Translation. The most popular approach to MT system combination involves building confusion networks from the outputs of different MT systems and decoding them to generate new translations (Rosti et al., 2007; Karakos et al., 2008; He et al., 2008; Xu et al., 2011). Other researchers explored the idea of re-ranking the n-best output of MT systems using different types of syntactic models (Och et al., 2004; Hasan et al., 2006; Ma and McKeown, 2013). While most researchers use target language features in training their re-rankers, others considered source language features (Ma and McKeown, 2013).

Most MT system combination work uses MT systems employing different techniques to train on the same data. However, in this paper, we use the same MT algorithms for training, tuning, and testing, but vary the training data, specifically in terms of the degree of source language dialectness. Our approach runs a classifier trained only on source language features to decide which system should translate each sentence in the test set, which means that each sentence goes through one MT system only. Since we do not combine the output of the MT systems on the phrase level, we call our approach "*system selection*" to avoid confusion.

3 Machine Translation Experiments

In this section, we present our MT experimental setup and the four baseline systems we built, and we evaluate their performance and the potential of their combination. In the next section we present and evaluate the system selection approach.

MT Tools and Settings. We use the open-source Moses toolkit (Koehn et al., 2007) to build four Arabic-English phrase-based statistical machine translation systems (SMT). Our systems use a standard phrase-based architecture. The parallel corpora are word-aligned using GIZA++ (Och and Ney, 2003). The language model for our systems is trained on English Gigaword (Graff and Cieri, 2003). We use SRILM Toolkit (Stolcke, 2002) to build a 5-gram language model with modified

Kneser-Ney smoothing. Feature weights are tuned to maximize BLEU on tuning sets using Minimum Error Rate Training (Och, 2003). Results are presented in terms of BLEU (Papineni et al., 2002). All evaluation results are case *insensitive*. The English data is tokenized using simple punctuation-based rules. The MSA portion of the Arabic side is segmented according to the Arabic Treebank (ATB) tokenization scheme (Maamouri et al., 2004; Sadat and Habash, 2006) using the MADA+TOKAN morphological analyzer and tokenizer v3.1 (Roth et al., 2008), while the DA portion is ATB-tokenized with MADA-ARZ (Habash et al., 2013). The Arabic text is also Alif/Ya normalized. For more details on processing Arabic, see (Habash, 2010).

MT Train/Tune/Test Data. We have two parallel corpora. The first is a *DA-English* corpus of 5M tokenized words of Egyptian (~3.5M) and Levantine (~1.5M). This corpus is part of BOLT data. The second is an *MSA-English* corpus of 57M tokenized words obtained from several LDC corpora (10 times the size of the DA-English data). We work with eight standard MT test sets: three MSA sets from NIST MTEval with four references (MT06, MT08, and MT09), four Egyptian sets from LDC BOLT data with two references (EgyDevV1, EgyDevV2, EgyDevV3, and EgyTestV2), and one Levantine set from BBN (Zbib et al., 2012) with one reference which we split into LevDev and LevTest. We used MT08 and EgyDevV3 to tune SMT systems while we divided the remaining sets among classifier training data (5,562 sentences), dev (1,802 sentences) and blind test (1,804 sentences) sets to ensure each of these new sets has a variety of dialects and genres (weblog and newswire).

MT Systems. We build four MT systems.

(1) **DA-Only.** This system is trained on the DA-English data and tuned on EgyDevV3.

(2) **MSA-Only.** This system is trained on the MSA-English data and tuned on MT08.

(3) **DA+MSA.** This system is trained on the combination of both corpora (resulting in 62M tokenized² words on the Arabic side) and tuned on

²Since the *DA+MSA* system is intended for DA data and DA morphology, as far as tokenization is concerned, is more complex, we tokenized the training data with dialect awareness (DA with MADA-ARZ and MSA with MADA) since MADA-ARZ does a lot better than MADA on DA (Habash et al., 2013). Tuning and Test data, however, are tokenized by MADA-ARZ since we do not assume any knowledge of the dialect of a test sentence.

EgyDevV3.

(4) **MSA-Pivot.** This MSA-pivoting system uses Salloum and Habash (2013)’s DA-MSA MT system followed by an Arabic-English SMT system which is trained on both corpora augmented with the DA-English where the DA side is preprocessed with the same DA-MSA MT system then tokenized with MADA-ARZ. The result is 67M tokenized words on the Arabic side. EgyDevV3 was similarly preprocessed with the DA-MSA MT system and MADA-ARZ and used for tuning the system parameters. Test sets are similarly preprocessed before decoding with the SMT system.

Baseline MT System Results. We report the results of our dev set on the four MT systems we built in Table 1. The *MSA-Pivot* system produces the best singleton result among all systems. All differences in BLEU scores between the four systems are statistically significant above the 95% level. Statistical significance is computed using paired bootstrap re-sampling (Koehn, 2004).

System Name	Training Data (TD)				BLEU
	DA-En	MSA-En	DA ^T -En	TD Size	
1. <i>DA-Only</i>	5M			5M	26.6
2. <i>MSA-Only</i>		57M		57M	32.7
3. <i>DA+MSA</i>	5M	57M		62M	33.6
4. <i>MSA-Pivot</i>	5M	57M	5M	67M	33.9
Oracle System Selection					39.3

Table 1: Results from the baseline MT systems and their oracle system selection. The training data used in different MT systems are also indicated. DA^T (in the fourth column) is the DA part of the 5M word DA-En parallel data processed with the DA-MSA MT system.

Oracle System Selection. We also report in Table 1 an oracle system selection where we pick, for each sentence, the English translation that yields the best BLEU score. This oracle indicates that the upper bound for improvement achievable from system selection is 5.4% BLEU. Excluding different systems from the combination lowered the overall score between 0.9% and 1.8%, suggesting the systems are indeed complementary.

4 MT System Selection

The approach we take in this paper benefits from the techniques and conclusions of previous papers in that we build different MT systems similar to those discussed above but instead of trying to find which one is the best, we try to leverage the use of all of them by automatically deciding what sentences should go to which system. Our hypothesis

is that these systems complement each other in interesting ways where the combination of their selections could lead to better overall performance stipulating that our approach could benefit from the strengths while avoiding the weaknesses of each individual system.

4.1 Dialect ID Binary Classification

For baseline system selection, we use the classification decision of Elfardy and Diab (2013)’s sentence-level dialect identification system to decide on the target MT system. Since the decision is binary (DA or MSA) and we have four MT systems, we considered all possible configurations and determined empirically that the best configuration is to select *MSA-Only* for the MSA tag and *MSA-Pivot* for the DA tag. We do not report other configuration results due to space restrictions.

4.2 Feature-based Four-Class Classification

For our main approach, we train a four-class classifier to predict the target MT system to select for each sentence using only source-language features. We experimented with different classifiers in the Weka Data Mining Tool (Hall et al., 2009) for training and testing our system selection approach. The best performing classifier was Naive Bayes (with Weka’s default settings).

Training Data Class Labels. We run the 5,562 sentences of the classification training data through our four MT systems and produce sentence-level BLEU scores (with length penalty). We pick the name of the MT system with the highest BLEU score as the class label for that sentence. When there is a tie in BLEU scores, we pick the system label that yields better overall BLEU scores from the systems tied.

Training Data Source-Language Features. We use two sources of features extracted from untokenized sentences to train our four-class classifiers: *basic* and *extended features*.

A. Basic Features

These are the same set of features that were used by the dialect ID tool together with the class label generated by this tool.

i. Token-Level Features. These features rely on language models, MSA and Egyptian morphological analyzers and a Highly Dialectal Egyptian lexicon to decide whether each word is MSA, Egyptian, Both, or Out of Vocabulary.

ii. Perplexity Features. These are two features that measure the perplexity of a sentence against

two language models: MSA and Egyptian.

iii. Meta Features. Features that do not directly relate to the dialectalness of words in the given sentence but rather estimate how informal the sentence is and include: percentage of tokens, punctuation, and Latin words, number of tokens, average word length, whether the sentence has any words that have word-lengthening effects or not, whether the sentence has any diacritized words or not, whether the sentence has emoticons or not, whether the sentence has consecutive repeated punctuation or not, whether the sentence has a question mark or not, and whether the sentence has an exclamation mark or not.

iv. The Dialect-Class Feature. We run the sentence through the Dialect ID binary classifier and we use the predicted class label (DA or MSA) as a feature in our system. Since the Dialect ID system was trained on a different data set, we think its decision may provide additional information to our classifiers.

B. Extended Features

We add features extracted from two sources.

i. MSA-Pivoting Features. Salloum and Habash (2013) DA-MSA MT system produces intermediate files used for diagnosis or debugging purposes. We exploit one file in which the system identifies (or, "selects") dialectal words and phrases that need to be translated to MSA. We extract confidence indicating features. These features are: sentence length (in words), percentage of selected words and phrases, number of selected words, number of selected phrases, number of words morphologically selected as dialectal by a mainly Levantine morphological analyzer, number of words selected as dialectal by the tool's DA-MSA lexicons, number of OOV words against the *MSA-Pivot* system training data, number of words in the sentences that appeared less than 5 times in the training data, number of words in the sentences that appeared between 5 and 10 times in the training data, number of words in the sentences that appeared between 10 and 15 times in the training data, number of words that have spelling errors and corrected by this tool (e.g., word-lengthening), number of punctuation marks, and number of words that are written in Latin script.

ii. MT Training Data Source-Side LM Perplexity Features. The second set of features uses perplexity against language models built from the source-side of the training data of each of the four

baseline systems. These four features may tell the classifier which system is more suitable to translate a given sentence.

4.3 System Selection Evaluation

Development Set. The first part of Table 2 repeats the best baseline system and the four-system oracle combination from Table 1 for convenience. The third row shows the result of running our system selection baseline that uses the Dialect ID binary decision on the Dev set sentences to decide on the target MT system. It improves over the best single system baseline (*MSA-Pivot*) by a statistically significant 0.5% BLEU. Crucially, we should note that this is a deterministic process.

System	BLEU	Diff.
Best Single MT System Baseline	33.9	0.0
Oracle	39.3	5.4
Dialect ID Binary Selection Baseline	34.4	0.5
Four-Class Classification		
Basic Features	35.1	1.2
Extended Features	34.8	0.9
Basic + Extended Features	35.2	1.3

Table 2: Results of baselines and system selection systems on the Dev set in terms of BLEU. The best single MT system baseline is *MSA-Pivot*.

The second part of Table 2 shows the results of our four-class Naive Bayes classifiers trained on the classification training data we created. The first column shows the source of sentence level features employed. As mentioned earlier, we use the Basic features alone, the Extended features alone, and then their combination. The classifier that uses both feature sources simultaneously as feature vectors is our best performer. It improves over our best baseline single MT system by 1.3% BLEU and over the Dialect ID Binary Classification system selection baseline by 0.8% BLEU. Improvements are statistically significant.

System	BLEU	Diff.
<i>DA-Only</i>	26.6	
<i>MSA-Only</i>	30.7	
<i>DA+MSA</i>	32.4	
<i>MSA-Pivot</i>	32.5	
<i>Four-System Oracle Combination</i>	38.0	5.5
Best Dialect ID Binary Classifier	32.9	0.4
Best Classifier: Basic + Extended Features	33.5	1.0

Table 3: Results of baselines and system selection systems on the Blind test set in terms of BLEU.

Blind Test Set. Table 3 shows the results on our Blind Test set. The first part of the table shows the results of our four baseline MT systems. The systems have the same rank as on the Dev set and

System	All	Dialect	MSA
<i>DA-Only</i>	26.6	19.3	33.2
<i>MSA-Only</i>	32.7	14.7	50.0
<i>DA+MSA</i>	33.6	19.4	46.3
<i>MSA-Pivot</i>	33.9	19.6	46.4
<i>Four-System Oracle Combination</i>	39.3	24.4	52.1
Best Performing Classifier	35.2	19.8	50.0

Table 4: Dialect breakdown of performance on the Dev set for our best performing classifier against our four baselines and their oracle combination. Our classifier does not know of these subsets, it runs on the set as a whole; therefore, we repeat its results in the second column for convenience.

MSA-Pivot is also the best performer. The differences in BLEU are statistically significant. The second part shows the four-system oracle combination which shows a 5.5% BLEU upper bound on improvements. The third part shows the results of the Dialect ID Binary Classification which improves by 0.4% BLEU. The last row shows the four-class classifier results which improves by 1.0% BLEU over the best single MT system baseline and by 0.6% BLEU over the Dialect ID Binary Classification. Results on the Blind Test set are consistent with the Dev set results.

5 Discussion and Error Analysis

DA versus MSA Performance. In Table 4, column **All** illustrates the results over the entire Dev set, while columns **DA** and **MSA** show system performance on the DA and MSA subsets of the Dev set, respectively. The best single baseline MT system for DA is *MSA-Pivot* has a large room for improvement given the oracle upper bound (4.8% BLEU absolute). However, our best system selection approach improves over *MSA-Pivot* by a small margin of 0.2% BLEU absolute only, albeit a statistically significant improvement. The MSA column oracle shows a smaller improvement of 2.1% BLEU absolute over the best single *MSA-Only* MT system. Furthermore, when translating MSA with our best system selection performer we get the same results as the best baseline MT system for MSA even though our system does not know the dialect of the sentences a priori. If we consider the breakdown of the performance in our best overall (33.9% BLEU) single baseline MT system (*MSA-Pivot*), we observe that the performance on MSA is about 3.6% absolute BLEU points below our best results; this suggests that most of the system selection gain over the best single baseline is on MSA selection.

Manual Error Analysis. We performed manual error analysis on a Dev set sample of 250 sen-

tences distributed among the different dialects and genres. Our best performing classifier selected the best system in 48% of the DA cases and 52% of the MSA cases. We did a detailed manual error analysis for the cases where the classifier failed to predict the best MT system. The sources of errors we found cover 89% of the cases. In 21% of the error cases, our classifier predicted a better translation than the one considered gold by BLEU due to BLEU bias, e.g., severe sentence-level length penalty due to an extra punctuation in a short sentence. Also, 3% of errors are due to bad references, e.g., a dialectal sentence in an MSA set that the human translators did not understand.

A group of error sources resulted from MSA sentences classified correctly as *MSA-Only*; however, one of the other three systems produced better translations for two reasons. First, since the MSA training data is from an older time span than the DA data, 10% of errors are due to MSA sentences that use recent terminology (e.g., Egyptian revolution 2011: places, politicians, etc.) that appear in the DA training data. Also, web writing styles in MSA sentences such as blog style (e.g., rhetorical questions), blog punctuation marks (e.g., "...", "???!"), and formal MSA forum greetings resulted in 23%, 16%, and 6% of the cases, respectively.

Finally, in 10% of the cases our classifier is confused by a code-switched sentence, e.g., a dialectal proverb in an MSA sentence or a weak MSA literal translation of dialectal words and phrases. Some of these cases may be solved by adding more features to our classifier, e.g., blog style writing features, while others need a radical change to our technique such as word and phrase level dialect identification for MT system combination of code-switched sentences.

6 Conclusion and Future Work

We presented a sentence-level classification approach for MT system selection for diglossic languages. We got a 1.0% BLEU improvement over the best baseline single MT system. In the future we plan to add more training data to see the effect on the accuracy of system selection. We plan to give different weights to different training examples based on the drop in BLEU score the example can cause if classified incorrectly. We also plan to explore confusion-network combination and re-ranking techniques based on target language features.

References

- Murat Akbacak, Dimitra Vergyri, Andreas Stolcke, Nicolas Scheffer, and Arindam Mandal. 2011. Effective arabic dialect classification using diverse phonotactic models. In *INTERSPEECH*, volume 11, pages 737–740.
- Fadi Biadisy, Julia Hirschberg, and Nizar Habash. 2009. Spoken arabic dialect identification using phonotactic modeling. In *Proceedings of the Workshop on Computational Approaches to Semitic Languages at the meeting of the European Association for Computational Linguistics (EACL), Athens, Greece*.
- Heba Elfardy and Mona Diab. 2013. Sentence Level Dialect Identification in Arabic. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics (ACL-13)*, Sofia, Bulgaria.
- Heba Elfardy, Mohamed Al-Badrashiny, and Mona Diab. 2013. Code switch point detection in arabic. In *Proceedings of the 18th International Conference on Application of Natural Language to Information Systems (NLDB2013)*, MediaCity, UK.
- Charles F Ferguson. 1959. Diglossia. *Word*, 15(2):325–340.
- David Graff and Christopher Cieri. 2003. English Gigaword, LDC Catalog No.: LDC2003T05. Linguistic Data Consortium, University of Pennsylvania.
- Nizar Habash, Ryan Roth, Owen Rambow, Ramy Eskander, and Nadi Tomeh. 2013. Morphological Analysis and Disambiguation for Dialectal Arabic. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Atlanta, GA.
- Nizar Habash. 2010. *Introduction to Arabic Natural Language Processing*. Morgan & Claypool Publishers.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18.
- S. Hasan, O. Bender, and H. Ney. 2006. Reranking translation hypotheses using structural properties. In *EACL'06 Workshop on Learning Structured Information in Natural Language Applications*.
- Xiaodong He, Mei Yang, Jianfeng Gao, Patrick Nguyen, and Robert Moore. 2008. Indirect-hmm-based hypothesis alignment for combining outputs from machine translation systems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 98–107. Association for Computational Linguistics.
- Damianos Karakos, Jason Eisner, Sanjeev Khudanpur, and Markus Dreyer. 2008. Machine translation system combination using itg-based alignments. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 81–84. Association for Computational Linguistics.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Christopher Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Christopher Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July. Association for Computational Linguistics.
- Wei-Yun Ma and Kathleen McKeown. 2013. Using a supertagged dependency language model to select a good translation in system combination. In *Proceedings of NAACL-HLT*, pages 433–438.
- Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. In *NEMLAR Conference on Arabic Language Resources and Tools*, pages 102–109, Cairo, Egypt.
- F. J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Franz Josef Och. 2004. A smorgasbord of features for statistical machine translation. In *Meeting of the North American chapter of the Association for Computational Linguistics*.
- Franz Josef Och. 2003. Minimum Error Rate Training for Statistical Machine Translation. In *Proceedings of the 41st Annual Conference of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan.
- 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, PA.
- Antti-Veikko Rosti, Spyros Matsoukas, and Richard Schwartz. 2007. Improved word-level system combination for machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 312–319, Prague, Czech Republic, June. Association for Computational Linguistics.
- Ryan Roth, Owen Rambow, Nizar Habash, Mona Diab, and Cynthia Rudin. 2008. Arabic Morphological Tagging, Diacritization, and Lemmatization Using Lexeme Models and Feature Ranking. In *Proceedings of ACL-08: HLT, Short Papers*, pages 117–120, Columbus, Ohio.
- Fatiha Sadat and Nizar Habash. 2006. Combination of Arabic preprocessing schemes for statistical ma-

- chine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 1–8, Sydney, Australia, July. Association for Computational Linguistics.
- Hassan Sajjad, Kareem Darwish, and Yonatan Belinkov. 2013. Translating dialectal arabic to english. In *The 51st Annual Meeting of the Association for Computational Linguistics - Short Papers (ACL Short Papers 2013)*, Sofia, Bulgaria.
- Wael Salloum and Nizar Habash. 2011. Dialectal to Standard Arabic Paraphrasing to Improve Arabic-English Statistical Machine Translation. In *Proceedings of the First Workshop on Algorithms and Resources for Modelling of Dialects and Language Varieties*, pages 10–21, Edinburgh, Scotland.
- Wael Salloum and Nizar Habash. 2013. Dialectal Arabic to English Machine Translation: Pivoting through Modern Standard Arabic. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Atlanta, GA.
- Hassan Sawaf. 2010. Arabic dialect handling in hybrid machine translation. In *Proceedings of the Conference of the Association for Machine Translation in the Americas (AMTA)*, Denver, Colorado.
- Andreas Stolcke. 2002. SRILM an Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing*.
- Daguang Xu, Yuan Cao, and Damianos Karakos. 2011. Description of the jhu system combination scheme for wmt 2011. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 171–176. Association for Computational Linguistics.
- Omar F Zaidan and Chris Callison-Burch. 2011. The arabic online commentary dataset: an annotated dataset of informal arabic with high dialectal content. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-11)*, pages 37–41.
- Rabih Zbib, Erika Malchiodi, Jacob Devlin, David Stallard, Spyros Matsoukas, Richard Schwartz, John Makhoul, Omar F. Zaidan, and Chris Callison-Burch. 2012. Machine Translation of Arabic Dialects. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 49–59, Montréal, Canada, June. Association for Computational Linguistics.

RNN-based Derivation Structure Prediction for SMT

Feifei Zhai, Jiajun Zhang, Yu Zhou and Chengqing Zong

National Laboratory of Pattern Recognition

Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China

{ffzhai, jjzhang, yzhou, cqzong}@nlpr.ia.ac.cn

Abstract

In this paper, we propose a novel derivation structure prediction (DSP) model for SMT using recursive neural network (RNN). Within the model, two steps are involved: (1) phrase-pair vector representation, to learn vector representations for phrase pairs; (2) derivation structure prediction, to generate a bilingual RNN that aims to distinguish good derivation structures from bad ones. Final experimental results show that our DSP model can significantly improve the translation quality.

1 Introduction

Derivation structure is important for SMT decoding, especially for the translation model based on nested structures of languages, such as BTG (bracket transduction grammar) model (Wu, 1997; Xiong et al., 2006), hierarchical phrase-based model (Chiang, 2007), and syntax-based model (Galley et al., 2006; Marcu et al., 2006; Liu et al., 2006; Huang et al., 2006; Zhang et al., 2008; Zhang et al., 2011; Zhai et al., 2013). In general, *derivation structure* refers to the tuple that records the used translation rules and their compositions during decoding, just as Figure 1 shows.

Intuitively, a good derivation structure usually yields a good translation, while bad derivations always result in bad translations. For example in Figure 1, (a) and (b) are two different derivations for Chinese sentence “布什与沙龙举行了会谈”. Comparing the two derivations, (a) is more reasonable and yields a better translation. However, (b) wrongly translates phrase “与沙龙” to “and Sharon” and combines it with [布什;Bush] incorrectly, leading to a bad translation.

To explore the derivation structure’s potential on yielding good translations, in this paper, we propose a novel derivation structure prediction (DSP) model for SMT decoding.

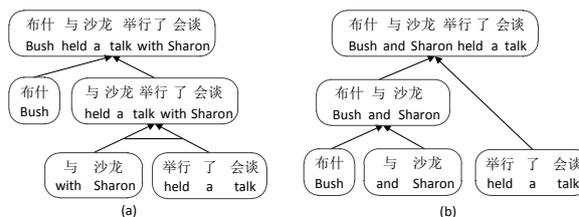


Figure 1: Two different derivation structures of BTG translation model. In the structure, leaf nodes denote the used translation rules. For each node, the first line is the source string, while the second line is its corresponding translation.

The proposed DSP model is built on recursive neural network (RNN). Within the model, two steps are involved: (1) phrase-pair vector representation, to learn vector representations for phrase pairs; (2) derivation structure prediction, to build a bilingual RNN that aims to distinguish good derivation structures from bad ones. Extensive experiments show that the proposed DSP model significantly improves the translation quality, and thus verify the effectiveness of derivation structure on indicating good translations.

We make the following contributions in this work:

- We propose a novel RNN-based model to do derivation structure prediction for SMT decoding. To our best knowledge, this is the first work on this issue in SMT community;
- In current work, RNN has only been verified to be useful on monolingual structure learning (Socher et al., 2011a; Socher et al., 2013). We go a step further, and design a bilingual RNN to represent the derivation structure;
- To train the RNN-based DSP model, we propose a max-margin objective that prefers gold derivations yielded by forced decoding to n-best derivations generated by the conventional BTG translation model.

2 The DSP Model

The basic idea of DSP model is to represent the derivation structure by RNN (Figure 2). Here, we build the DSP model for BTG translation model, which is naturally compatible with RNN. We believe that the DSP model is also beneficial to other translation models. We leave them as our future work.

2.1 Phrase-Pair Vector Representation

Phrase pairs, i.e., the used translation rules, are the leaf nodes of derivation structure. Hence, to represent the derivation structure by RNN, we need first to represent the phrase pairs. To do this, we use two unsupervised recursive autoencoders (RAE) (Socher et al., 2011b), one for the source phrase and the other for the target phrase. We call the unit of the two RAEs the **Leaf Node Network (LNN)**.

Using n -dimension word embedding, RAE can learn a n -dimension vector for any phrase. Meanwhile, RAE will build a binary tree for the phrase, as Figure 2 (in box) shows, and compute a reconstruction error to evaluate the vector. We use $E(T_{ph})$ to denote the reconstruction error given by RAE, where ph is the phrase and T_{ph} is the corresponding binary tree. In RAE, higher error corresponds to worse vector. More details can be found in (Socher et al., 2011b).

Given a phrase pair (sp, tp) , we can use LNN to generate two n -dimension vectors, representing sp and tp respectively. Then, we concatenate the two vectors directly, and get a vector $r \in \mathbb{R}^{2n}$ to represent phrase pair (sp, tp) (shown in Figure 2). The vector r is evaluated by combining the reconstruction error on both sides:

$$E(T_{sp}, T_{tp}) = \frac{1}{2} [E(T_{sp}) + E(T_{tp}) \cdot \frac{N_s}{N_t}] \quad (1)$$

where T_{sp} and T_{tp} are the binary trees for sp and tp . N_s and N_t denote the number of nodes in T_{sp} and T_{tp} . Note that in order to unify the errors on the two sides, we use ratio N_s/N_t to eliminate the influence of phrase length.

Then, according to Equation (1), we compute an LNN score to evaluate the vector of all phrase pairs, i.e., leaf nodes, in derivation d :

$$LNN(d) = - \sum_{(sp, tp)} E(T_{sp}, T_{tp}) \quad (2)$$

where (sp, tp) is the used phrase pair in derivation d . Obviously, the derivation with better phrase-pair representations will get a higher LNN score.

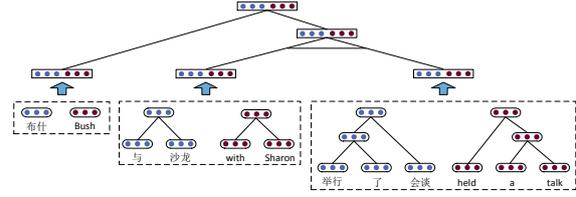


Figure 2: Illustration of DSP model, based on the derivation structure in Figure 1(a).

The LNN score will serve as part of the DSP model for predicting good derivation structures.

2.2 Derivation Structure Prediction

Using the vector representations of phrase pairs, we then build a **Derivation Structure Network (DSN)** for prediction (Figure 2).

In DSN, the derivation structure is represented by repeatedly applying **unit neural network (UNN, Figure 3)** at each non-leaf node. The UNN receives two node vectors $r_1 \in \mathbb{R}^{2n}$ and $r_2 \in \mathbb{R}^{2n}$ as input, and induces a vector $p \in \mathbb{R}^{2n}$ to represent the parent node.

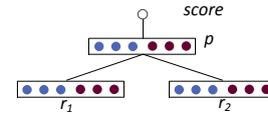


Figure 3: The unit neural network used in DSN.

For example, in Figure 2, node [与 沙龙; with Sharon] serves as the first child with vector r_1 , and node [举行了会谈; held a talk] as the second child with vector r_2 . The parent node vector p , representing [与 沙龙 举行了会谈; held a talk with Sharon], is computed by merging r_1 and r_2 :

$$p = f(W_{UNN}[r_1; r_2] + b_{UNN}) \quad (3)$$

where $[r_1; r_2] \in \mathbb{R}^{4n \times 1}$ is the concatenation of r_1 and r_2 , $W_{UNN} \in \mathbb{R}^{2n \times 4n}$ and $b_{UNN} \in \mathbb{R}^{2n \times 1}$ are the network's parameter weight matrix and bias term respectively. We use $\tanh(\cdot)$ as function f .

Then, we compute a local score using a simple inner product with a row vector $W_{UNN}^{score} \in \mathbb{R}^{1 \times 2n}$:

$$s(p) = W_{UNN}^{score} \cdot p \quad (4)$$

The score measures how well the two child nodes r_1 and r_2 are merged into the parent node p .

As we all know, in BTG derivations, we have two different ways to merge translation candidates, *monotone* or *inverted*, meaning that we

merge two candidates in a monotone or inverted order. We believe that different merging order (monotone or inverted) needs different UNN. Hence, we keep two different ones in DSN, one for monotone order (with parameter W_{mono} , b_{mono} , and W_{mono}^{score}), and the other for inverted (with parameter W_{inv} , b_{inv} , and W_{inv}^{score}). The idea is that the merging order of the two candidates will determine which UNN will be used to generate their parent’s vector and compute the score in Equation (4). Using a set of gold derivations, we can train the network so that correct order will receive a high score by Equation (4) and incorrect one will receive a low score.

Thus, when we merge the candidates of two adjacent spans during BTG-based decoding, the local score in Equation (4) is useful in two aspects: (1) for the same merging order, it evaluates how well the two candidates are merged; (2) for the different order, it compares the candidates generated by monotone order and inverted order.

Further, to assess the entire derivation structure, we apply UNN to each node recursively, until the root node. The final score utilized for derivation structure prediction is the sum of all local scores:

$$DSN(d) = \sum_p s(p) \quad (5)$$

where d denotes the derivation structure and p is the non-leaf node in d . Obviously, by this score, we can easily assess different derivations. Good derivations will get higher scores while bad ones will get lower scores.

Li et al. (2013) presented a network to predict how to merge translation candidates, in monotone or inverted order. Our DSN differs from Li’s work in two points. For one thing, DSN can not only predict how to merge candidates, but also evaluate whether two candidates should be merged. For another, DSN focuses on the entire derivation structure, rather than only the two candidates for merging. Therefore, the translation decoder will pursue good derivation structures via DSN. Actually, Li’s work can be easily integrated into our work. We leave it as our future work.

3 Training

In this section, we present the method of training the DSP model. The parameters involved in this process include: word embedding, parameters of the two unsupervised RAEs in LNN, and parameters in DSN.

3.1 Max-Margin Framework

In DSP model, our goal is to assign higher scores to gold derivations, and lower scores to bad ones. To reach this goal, we adopt a max-margin framework (Socher et al., 2010; Socher et al., 2011a; Socher et al., 2013) for training.

Specifically, suppose we have a training data like $(u_i, \mathcal{G}(u_i), \mathcal{A}(u_i))$, where u_i is the input source sentence, $\mathcal{G}(u_i)$ is the **gold derivation set** containing all gold derivations of u_i ¹, and $\mathcal{A}(u_i)$ is the **possible derivation set** that contains all possible derivations of u_i . We want to minimize the following regularized risk function:

$$J(\theta) = \frac{1}{N} \sum_{i=1}^N R_i(\theta) + \frac{\lambda}{2} \|\theta\|^2, \text{ where} \\ R_i(\theta) = \max_{\hat{d} \in \mathcal{A}(u_i)} \left(s(\theta, u_i, \hat{d}) + \Delta(\hat{d}, \mathcal{G}(u_i)) \right) \\ - \max_{d \in \mathcal{G}(u_i)} \left(s(\theta, u_i, d) \right) \quad (6)$$

Here, θ is the model parameter. $s(\theta, u_i, d)$ is the DSP score for sentence u_i ’s derivation d . It is computed by summing LNN score (Equation (2)) and DSN score (Equation (5)):

$$s(\theta, u, d) = LNN_{\theta}(d) + DSN_{\theta}(d) \quad (7)$$

$\Delta(\hat{d}, \mathcal{G}(u_i))$ is the structure loss margin, which penalizes derivation \hat{d} more if it deviates more from gold derivations. It is formulated as:

$$\Delta(\hat{d}, \mathcal{G}(u_i)) \\ = \sum_{\pi \in \hat{d}} \alpha_s \delta\{\pi \notin \mathcal{G}(u_i)\} + \alpha_t Dist(y(\hat{d}), ref) \quad (8)$$

The margin includes two parts. For the first part, π is the source span in derivation \hat{d} , $\delta\{\cdot\}$ is an indicator function. We use the first part to count the number of source spans in derivation \hat{d} , but not in gold derivations. The second part is for target side. $Dist(y(\hat{d}), ref)$ computes the edit-distance between the translation result $y(\hat{d})$ defined by derivation \hat{d} and the reference translation ref . Obviously, this margin can effectively estimate the difference between derivation \hat{d} and gold derivations, both on source side and target side. Note that α_s and α_t are only two hyperparameters for scaling. They are independent of each other, and we set $\alpha_s = 0.1$ and $\alpha_t = 0.1$ respectively.

¹We investigate the general case here and suppose that one sentence could have several different gold derivations. In the experiment, we only use one gold derivation for simple implementation.

3.2 Learning

As the risk function, Equation (6) is not differentiable. We train the model via the subgradient method (Ratliff et al., 2007; Socher et al., 2013). For parameter θ , the subgradient of $J(\theta)$ is:

$$\frac{\partial J}{\partial \theta} = \frac{1}{N} \sum_i \frac{\partial s(\theta, u_i, \hat{d}_m)}{\partial \theta} - \frac{\partial s(\theta, u_i, d_m)}{\partial \theta} + \lambda \theta$$

where \hat{d}_m is the derivation with the highest DSP score, and d_m denotes the gold derivation with the highest DSP score. We adopt the diagonal variant of AdaGrad (Duchi et al., 2011; Socher et al., 2013) to minimize the risk function for training.

3.3 Training Instances Collection

In order to train the model, we need to collect the gold derivation set $\mathcal{G}(u_i)$ and possible derivation set $\mathcal{A}(u_i)$ for input sentence u_i .

For $\mathcal{G}(u_i)$, we define it by *force decoding derivation (FDD)*. Basically, FDD refers to the derivation that produces the exact reference translation (single reference in our training data). For example, since “Bush held a talk with Sharon” is the reference of test sentence “布什与沙龙举行了会谈”, then Figure 1(a) is one of the FDDs. As FDD can produce reference translation, we believe that FDD is of high quality, and take them as gold derivations for training.

For $\mathcal{A}(u_i)$, it should contain all possible derivations of u_i . However, it is too difficult to obtain all derivations. Thus, we use n-best derivations of SMT decoding to simulate the complete derivation space, and take them as the derivations in $\mathcal{A}(u_i)$.

4 Integrating the DSP Model into SMT

To integrate the DSP model into decoding, we take it (named DSP feature) as one of the features in the log-linear framework of SMT. During decoding, the DSP feature is distributed to each node in the derivation structure. For the leaf node, the score in Equation (2), i.e., LNN score, serves as the feature. For the non-leaf node, Equation (4) plays the role. In order to give positive feature value to the log-linear framework (for logarithm), we normalize the DSP scores to $[0,1]$ during decoding. Due to the length limit, we ignore the specific normalization methods here. We just preform some simple transformations (such as adding a constant, computing reciprocal), and convert the scores proportionally to $[0,1]$ at last.

5 Experiments

5.1 Experimental Setup

To verify the effectiveness of our DSP model, we perform experiments on Chinese-to-English translation. The training data contains about 2.1M sentence pairs with about 27.7M Chinese words and 31.9M English words². We train a 5-gram language model by the Xinhua portion of Gigaword corpus and the English part of the training data. We obtain word alignment by GIZA++, and adopt the grow-diag-final-and strategy to generate the symmetric alignment. We use NIST MT 2003 data as the development set, and NIST MT04-08³ as the test set. We use MERT (Och, 2004) to tune parameters. The translation quality is evaluated by case-insensitive BLEU-4 (Papineni et al., 2002). The statistical significance test is performed by the re-sampling approach (Koehn, 2004). The baseline system is our in-house BTG system (Wu, 1997; Xiong et al., 2006; Zhang and Zong, 2009).

To train the DSP model, we first use Word2Vec⁴ toolkit to pre-train the word embedding on large-scale monolingual data. The used monolingual data contains about 1.06B words for Chinese and 1.12B words for English. The dimensionality of our vectors is 50. The detailed training process is as follows:

(1) Using the BTG system to perform force decoding on FBIS part of the bilingual training data⁵, and collect the sentences succeeded in force decoding (86,902 sentences in total)⁶. We then collect the corresponding force decoding derivations as gold derivations. Here, we only use the best force decoding derivation for simple implementation. In future, we will try to use multiple force decoding derivations for training.

(2) Collecting the bilingual phrases in the leaf nodes of gold derivations. We train LNN by these phrases via L-BFGS algorithm. Finally, we get 351,448 source phrases to train the source side RAE and 370,948 target phrases to train the target side RAE.

²LDC category number : LDC2000T50, LDC2002E18, LDC2003E07, LDC2004T07, LDC2005T06, LDC2002L27, LDC2005T10 and LDC2005T34.

³For MT06 and MT08, we only use the part of news data.

⁴<https://code.google.com/p/word2vec/>

⁵Here we only use the high quality corpus FBIS to guarantee the quality of force decoding derivation.

⁶Many sentence pairs fail in forced decoding due to many reasons, such as reordering limit, noisy alignment, and phrase length limit (Yu et al., 2013).

(3) Decoding the 86902 sentences by the BTG system to get n-best translations and corresponding derivations. The n-best derivations are used to simulate the entire derivation space. We retain at most 200-best derivations for each sentence.

(4) Leveraging force decoding derivations and n-best derivations to train the DSP model. Note that all parameters, including word embedding and parameters in LNN and DSN, are tuned together in this step. It takes about 15 hours to train the entire network using a 16-core, 2.9 GHz Xeon machine.

5.2 Experimental Results

We compare baseline BTG system and the DSP-augmented BTG system in this section. The final translation results are shown in Table 1.

After integrating the DSP model into BTG system, we get significant improvement on all test sets, about 1.0 BLEU points over BTG system on average. This comparison strongly demonstrates that our DSP model is useful and will be a good complement to current translation models.

Systems	BLEU(%)				
	MT04	MT05	MT06	MT08	Aver
BTG	36.91	34.69	33.83	27.17	33.15
BTG+DSP	37.41	35.77	35.08	28.42	34.17

Table 1: Final translation results. **Bold numbers** denote that the result is significantly better than baseline BTG system ($p < 0.05$). Column “Aver” gives the average BLEU points of the 4 test sets.

To have a better intuition for the effectiveness of our DSP model, we give a case study in Figure 4. It depicts two derivations built by BTG system and BTG+DSP system respectively.

From Figure 4(b), we can see that BTG system yields a bad translation due to the bad derivation structure. In the figure, BTG system makes three mistakes. It attaches candidates [成就; achievements], [所达到的; has reached] and [新加坡; singapore] to the big candidate [不能被当做理所当然; cannot be regarded as a natural]. Consequently, the noun phrase “新加坡所达到的成就” is translated separately, rather than as a whole, leading to a bad translation.

Differently, the DSP model is designed for predicting good derivations. In Figure 4(c), the used translation rules are actually similar to Figure 4(b). However, under a better guidance to build good derivation structure, BTG+DSP system generates a much better translation result than BTG system.

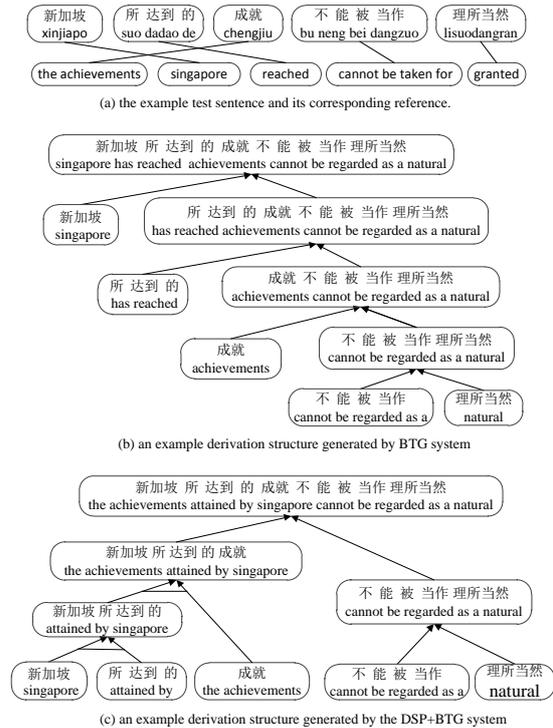


Figure 4: Different derivation structures.

6 Conclusion

In this paper, we explored the method of derivation structure prediction for SMT. To fulfill this task, we have made several major efforts as follows:

- (1) We propose a novel derivation structure prediction model based on RNN, including two close and interactive parts: LNN and DSN.
- (2) We extend monolingual RNN to bilingual RNN to represent the derivation structure.
- (3) We train LNN and DSN by derivations from force decoding. In this way, the DSP model learns a preference to good derivation structures.

Experimental results show that the proposed DSP model improves the translation performance significantly. By this, we verify the effectiveness of derivation structure on indicating good translations. We believe that our work will shed new lights to SMT decoding.

Acknowledgement

We would like to thank the three anonymous reviewers for their valuable comments and suggestions. The research work has been partially funded by the Natural Science Foundation of China under Grant No. 61333018 and 61303181, and the Key Project of Knowledge Innovation Program of Chinese Academy of Sciences.

References

- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.
- John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.
- Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968, Sydney, Australia, July. Association for Computational Linguistics.
- Liang Huang, Kevin Knight, and Aravind Joshi. 2006. A syntax-directed translator with extended domain of locality. In *Proceedings of AMTA*.
- Peng Li, Yang Liu, and Maosong Sun. 2013. Recursive autoencoders for itg-based translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616, Sydney, Australia, July. Association for Computational Linguistics.
- Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. Spmt: Statistical machine translation with syntactified target language phrases. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 44–52, Sydney, Australia, July. 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.
- Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. 2007. (online) subgradient methods for structured prediction. In *Eleventh International Conference on Artificial Intelligence and Statistics (AISTats)*.
- Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*.
- Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. 2011a. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 129–136.
- Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161.
- Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. 2013. Parsing with compositional vector grammars. In *Proceedings of ACL*.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational linguistics*, 23(3):377–403.
- Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of ACL-COLING*, pages 505–512.
- Heng Yu, Liang Huang, Haitao Mi, and Kai Zhao. 2013. Max-violation perceptron and forced decoding for scalable MT training. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1112–1123, Seattle, Washington, USA, October. Association for Computational Linguistics.
- Feifei Zhai, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2013. Unsupervised tree induction for tree-based translation. *Transactions of Association for Computational Linguistics(TACL)*, pages 291–300.
- Jiajun Zhang and Chengqing Zong. 2009. A framework for effectively integrating hard and soft syntactic rules into phrase based translation. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation*, pages 579–588, Hong Kong, December. City University of Hong Kong.
- Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In *Proceedings of ACL-08: HLT*, pages 559–567, Columbus, Ohio, June. Association for Computational Linguistics.
- Jiajun Zhang, Feifei Zhai, and Chengqing Zong. 2011. Augmenting string-to-tree translation models with fuzzy use of source-side syntax. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 204–215, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Hierarchical MT Training using Max-Violation Perceptron

Kai Zhao[†]

Liang Huang[†]

Haitao Mi[‡]

Abe Ittycheriah[‡]

[†]Graduate Center & Queens College
City University of New York

{kzhao@gc, huang@cs.qc}.cuny.edu

[‡]T. J. Watson Research Center
IBM

{hmi, abei}@us.ibm.com

Abstract

Large-scale discriminative training has become promising for statistical machine translation by leveraging the huge training corpus; for example the recent effort in phrase-based MT (Yu et al., 2013) significantly outperforms mainstream methods that only train on small tuning sets. However, phrase-based MT suffers from limited reorderings, and thus its training can only utilize a small portion of the bitext due to the distortion limit. To address this problem, we extend Yu et al. (2013) to syntax-based MT by generalizing their latent variable “violation-fixing” perceptron from graphs to hypergraphs. Experiments confirm that our method leads to up to +1.2 BLEU improvement over mainstream methods such as MERT and PRO.

1 Introduction

Many natural language processing problems including part-of-speech tagging (Collins, 2002), parsing (McDonald et al., 2005), and event extraction (Li et al., 2013) have enjoyed great success using large-scale discriminative training algorithms. However, a similar success on machine translation has been elusive, where the mainstream methods still tune on small datasets.

What makes large-scale MT training so hard then? After numerous attempts by various researchers (Liang et al., 2006; Watanabe et al., 2007; Arun and Koehn, 2007; Blunsom et al., 2008; Chiang et al., 2008; Flanigan et al., 2013; Green et al., 2013), the recent work of Yu et al. (2013) finally reveals a major reason: it is the vast amount of (inevitable) search errors in MT decoding that astray learning. To alleviate this problem, their work adopts the theoretically-motivated framework of violation-fixing perceptron (Huang et al., 2012) tailed for inexact search, yielding great results on phrase-based MT (outperforming

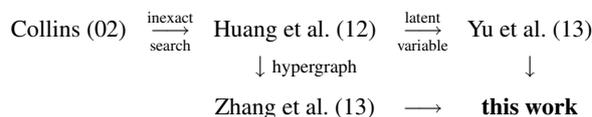


Figure 1: Relationship with previous work.

small-scale MERT/PRO by a large margin for the first time). However, the underlying phrase-based model suffers from limited distortion and thus can only employ a small portion (about 1/3 in their Ch-En experiments) of the bitext in training.

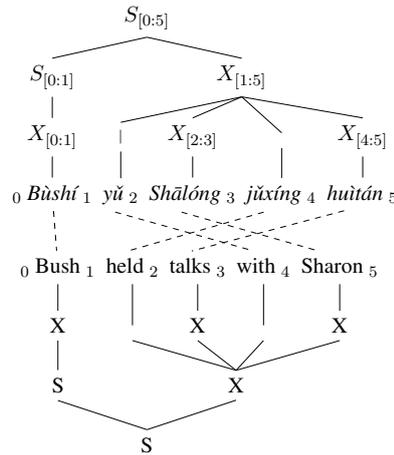
To better utilize the large training set, we propose to generalize from phrase-based MT to syntax-based MT, in particular the hierarchical phrase-based translation model (HIERO) (Chiang, 2005), in order to exploit sentence pairs beyond the expressive capacity of phrase-based MT.

The key challenge here is to extend the latent variable violation-fixing perceptron of Yu et al. (2013) to handle tree-structured derivations and translation hypergraphs. Luckily, Zhang et al. (2013) have recently generalized the underlying violation-fixing perceptron of Huang et al. (2012) from graphs to hypergraphs for bottom-up parsing, which resembles syntax-based decoding. We just need to further extend it to handle latent variables. We make the following contributions:

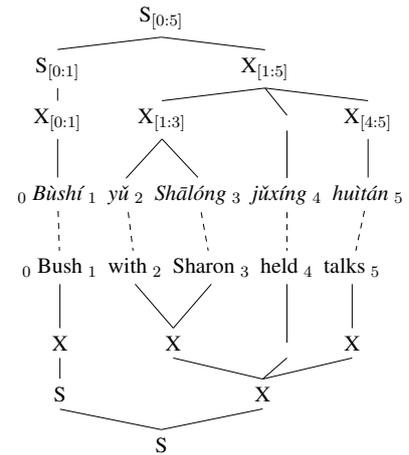
1. We generalize the latent variable violation-fixing perceptron framework to inexact search over hypergraphs, which subsumes previous algorithms for PBMT and bottom-up parsing as special cases (see Fig. 1).
2. We show that syntax-based MT, with its better handling of long-distance reordering, can exploit a larger portion of the training set, which facilitates sparse lexicalized features.
3. Experiments show that our training algorithm outperforms mainstream tuning methods (which optimize on small devsets) by +1.2 BLEU over MERT and PRO on FBIS.

id	rule
r_0	$S \rightarrow \langle X_{\square}, X_{\square} \rangle$
r_1	$S \rightarrow \langle S_{\square} X_{\square}, S_{\square} X_{\square} \rangle$
r_2	$X \rightarrow \langle Bùshí, Bush \rangle$
r_3	$X \rightarrow \langle Shānlóng, Sharon \rangle$
r_4	$X \rightarrow \langle huìtán, talks \rangle$
r_5	$X \rightarrow \langle yǔ X_{\square}, jǔxíng X_{\square}, \text{held } X_{\square} \text{ with } X_{\square} \rangle$
r_6	$X \rightarrow \langle yǔ Shānlóng, \text{with Sharon} \rangle$
r_7	$X \rightarrow \langle X_{\square}, jǔxíng X_{\square}, X_{\square} \text{ held } X_{\square} \rangle$

(a) HIERO rules

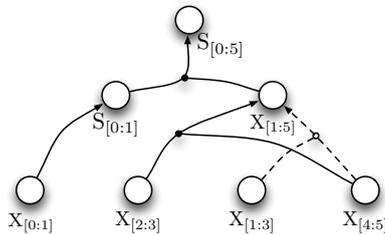


(b) gold derivation



(c) Viterbi derivation

Figure 2: An example of HIERO translation.

Figure 3: A $-$ LM hypergraph with two derivations: the gold derivation (Fig. 2b) in solid lines, and the Viterbi derivation (Fig. 2c) in dashed lines.

2 Review: Syntax-based MT Decoding

For clarity reasons we will describe HIERO decoding as a two-pass process, first without a language model, and then integrating the LM. This section mostly follows Huang and Chiang (2007).

In the first, $-$ LM phase, the decoder parses the source sentence using the source projection of the synchronous grammar (see Fig. 2 (a) for an example), producing a $-$ LM hypergraph where each node has a signature $N_{[i:j]}$, where N is the nonterminal type (either X or S in HIERO) and $[i:j]$ is the span, and each hyperedge e is an application of the translation rule $r(e)$ (see Figure 3).

To incorporate the language model, each node also needs to remember its target side boundary words. Thus a $-$ LM node $N_{[i:j]}$ is split into multiple $+$ LM nodes of signature $N_{[i:j]}^{a*b}$, where a and b are the boundary words. For example, with a bigram LM, $X_{[1:5]}^{\text{held*Sharon}}$ is a node whose translation starts with “held” and ends with “Sharon”.

More formally, the whole decoding process can be cast as a deductive system. Take the partial translation of “held talks with Sharon” in Figure 2

(b) for example, the deduction is

$$\frac{X_{[2:3]}^{\text{Sharon*Sharon}} : s_1 \quad X_{[4:5]}^{\text{talks*talks}} : s_2}{X_{[1:5]}^{\text{held*Sharon}} : s_1 + s_2 + s(r_5) + \lambda} \quad r_5,$$

where $s(r_5)$ is the score of rule r_5 , and the LM combo score λ is $\log P_{\text{lm}}(\text{talks} \mid \text{held})P_{\text{lm}}(\text{with} \mid \text{talks})P_{\text{lm}}(\text{Sharon} \mid \text{with})$.

3 Violation-Fixing Perceptron for HIERO

As mentioned in Section 1, the key to the success of Yu et al. (2013) is the adoption of violation-fixing perceptron of Huang et al. (2012) which is tailored for vastly inexact search. The general idea is to update somewhere in the middle of the search (where search error happens) rather than at the very end (standard update is often invalid). To adapt it to MT where many derivations can output the same translation (i.e., spurious ambiguity), Yu et al. (2013) extends it to handle latent variables which correspond to phrase-based derivations. On the other hand, Zhang et al. (2013) has generalized Huang et al. (2012) from graphs to hypergraphs for bottom-up parsing, which resembles HIERO decoding. So we just need to combine the two generalizing directions (latent variable and hypergraph, see Fig. 1).

3.1 Latent Variable Hypergraph Search

The key difference between bottom-up parsing and MT decoding is that in parsing the gold tree for each input sentence is unique, while in MT many derivations can generate the same reference translation. In other words, the gold derivation to update towards is a latent variable.

Here we formally define the latent variable “max-violation” perceptron over a hypergraph for MT training. For a given sentence pair $\langle x, y \rangle$, we denote $H(x)$ as the decoding hypergraph of HIERO without any pruning. We say $D \in H(x)$ if D is a full derivation of decoding x , and D can be derived from the hypergraph. Let $good(x, y)$ be the set of y -good derivations for $\langle x, y \rangle$:

$$good(x, y) \triangleq \{D \in H(x) \mid e(D) = y\},$$

where $e(D)$ is the translation from derivation D . We then define the set of y -good partial derivations that cover $x_{[i:j]}$ with root $N_{[i:j]}$ as

$$good_{N_{[i:j]}}(x, y) \triangleq \{d \in D \mid D \in good(x, y), \\ root(d) = N_{[i:j]}\}$$

We further denote the real decoding hypergraph with beam-pruning and cube-pruning as $H'(x)$. The set of y -bad derivations is defined as

$$bad_{N_{[i:j]}}(x, y) \triangleq \{d \in D \mid D \in H'(x, y), \\ root(d) = N_{[i:j]}, d \notin good_{N_{[i:j]}}(x, y)\}.$$

Note that the y -good derivations are defined over the *unpruned* whole decoding hypergraph, while the y -bad derivations are defined over the real decoding hypergraph with pruning.

The max-violation method performs the update where the model score difference between the incorrect Viterbi partial derivation and the best y -good partial derivation is maximal, by penalizing the incorrect Viterbi partial derivation and rewarding the y -good partial derivation.

More formally, we first find the Viterbi partial derivation d^- and the best y -good partial derivation d^+ for each $N_{[i:j]}$ group in the pruned +LM hypergraph:

$$d_{N_{[i:j]}}^+(x, y) \triangleq \underset{d \in good_{N_{[i:j]}}(x, y)}{\operatorname{argmax}} \mathbf{w} \cdot \Phi(x, d),$$

$$d_{N_{[i:j]}}^-(x, y) \triangleq \underset{d \in bad_{N_{[i:j]}}(x, y)}{\operatorname{argmax}} \mathbf{w} \cdot \Phi(x, d),$$

where $\Phi(x, d)$ is the feature vector for derivation d . Then it finds the group $N_{[i^*:j^*]}^*$ with the maximal score difference between the Viterbi derivation and the best y -good derivation:

$$N_{[i^*:j^*]}^* \triangleq \underset{N_{[i:j]}}{\operatorname{argmax}} \\ \mathbf{w} \cdot \Delta \Phi(x, d_{N_{[i:j]}}^+(x, y), d_{N_{[i:j]}}^-(x, y)),$$

and update as follows:

$$\mathbf{w} \leftarrow \mathbf{w} + \Delta \Phi(x, d_{N_{[i^*:j^*]}^+}^+(x, y), d_{N_{[i^*:j^*]}^-}^-(x, y)),$$

where $\Delta \Phi(x, d, d') \triangleq \Phi(x, d) - \Phi(x, d')$.

3.2 Forced Decoding for HIERO

We now describe how to find the gold derivations.¹ Such derivations can be generated in way similar to Yu et al. (2013) by using a language model tailored for forced decoding:

$$P_{forced}(q \mid p) = \begin{cases} 1 & \text{if } q = p + 1 \\ 0 & \text{otherwise} \end{cases},$$

where p and q are the indices of the boundary words in the reference translation. The +LM node now has signature $N_{[i:j]}^{p* q}$, where p and q are the indexes of the boundary words. If a boundary word does not occur in the reference, its index is set to ∞ so that its language model score will always be $-\infty$; if a boundary word occurs more than once in the reference, its -LM node is split into multiple +LM nodes, one for each such index.²

We have a similar deductive system for forced decoding. For the previous example, rule r_5 in Figure 2 (a) is rewritten as

$$X \rightarrow \langle y\check{u} X_{\square} j\check{u}x\check{i}ng X_{\square}, 1 X_{\square} 4 X_{\square} \rangle,$$

where 1 and 4 are the indexes for reference words “held” and “with” respectively. The deduction for $X_{[1:5]}$ in Figure 2 (b) is

$$\frac{X_{[2:3]}^{5*5} : s_1 \quad X_{[4:5]}^{2*3} : s_2}{X_{[1:5]}^{1*5} : s(r_5) + \lambda + s_1 + s_2} \quad r_5,$$

where $\lambda = \log \prod_{i \in \{1,3,4\}} P_{forced}(i + 1 \mid i) = 0$.

4 Experiments

Following Yu et al. (2013), we call our max-violation method MAXFORCE. Our implementation is mostly in Python on top of the `cdec` system (Dyer et al., 2010) via the `pycdec` interface (Chahuneau et al., 2012). In addition, we use minibatch parallelization of (Zhao and Huang,

¹We only consider *single* reference in this paper.

²Our formulation of index-based language model fixes a bug in the word-based LM of Yu et al. (2013) when a substring appears more than once in the reference (e.g. “the man...the man...”); thanks to Dan Gildea for pointing it out.

2013) to speedup perceptron training. We evaluate MAXFORCE for HIERO over two CH-EN corpora, IWSLT09 and FBIS, and compare the performance with vanilla n -best MERT (Och, 2003) from Moses (Koehn et al., 2007), Hypergraph MERT (Kumar et al., 2009), and PRO (Hopkins and May, 2011) from `cdec`.

4.1 Features Design

We use all the 18 *dense* features from `cdec`, including language model, direct translation probability $p(e|f)$, lexical translation probabilities $p_l(e|f)$ and $p_l(f|e)$, length penalty, counts for the source and target sides in the training corpus, and flags for the glue rules and pass-through rules.

For *sparse* features we use Word-Edges features (Charniak and Johnson, 2005; Huang, 2008) which are shown to be extremely effective in both parsing and phrase-based MT (Yu et al., 2013). We find that even simple Word-Edges features boost the performance significantly, and adding complex Word-Edges features from Yu et al. (2013) brings limited improvement and slows down the decoding. So in the following experiments we only use Word-Edges features consisting of combinations of English and Chinese words, and Chinese characters, and do not use word clusters nor word types. For simplicity and efficiency reasons, we also exclude all non-local features.

4.2 Datasets and Preprocessing

Our first corpus, IWSLT09, contains $\sim 30k$ short sentences collected from spoken language. IWSLT04 is used as development set in MAXFORCE training, and as tuning set for n -best MERT, Hypergraph MERT, and PRO. IWSLT05 is used as test set. Both IWSLT04 and IWSLT05 contain 16 references. We mainly use this corpus to investigate the properties of MAXFORCE.

The second corpus, FBIS, contains $\sim 240k$ sentences. NIST06 newswire is used as development set for MAXFORCE training, and as tuning set for all other tuning methods. NIST08 newswire is used as test set. Both NIST06 newswire and NIST08 newswire contain 4 references. We mainly use this corpus to demonstrate the performance of MAXFORCE in large-scale training.

For both corpora, we do standard tokenization, alignment and rule extraction using the `cdec` tools. In rule extraction, we remove all 1-count rules but keep the rules mapping from one Chinese word to one English word to help balancing

	sent.	words
phrase-based MT	32%	12%
HIERO	35%	30%
HIERO (all rules)	65%	55%

Table 1: Reachability comparison (on FBIS) between phrase-based MT reported in Yu et al. (2013) (without 1-count rules) and HIERO (with and without 1-count rules).

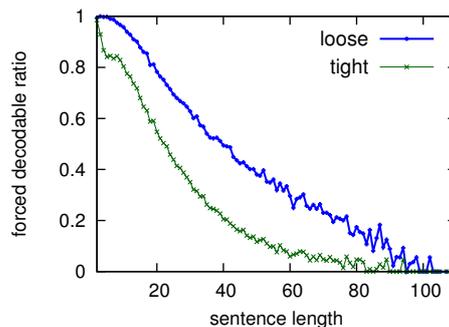


Figure 4: Reachability vs. sent. length on FBIS. See text below for “loose” and “tight”.

between overfitting and coverage. We use a trigram language model trained from the target sides of the two corpora respectively.

4.3 Forced Decoding Reachability

We first report the forced decoding reachability for HIERO on FBIS in Table 1. With the full rule set, 65% sentences and 55% words of the whole corpus are forced decodable in HIERO. After pruning 1-count rules, our forced decoding covers significantly more words than phrase-based MT in Yu et al. (2013). Furthermore, in phrase-based MT, most decodable sentences are very short, while in HIERO the lengths of decodable sentences are more evenly distributed.

However, in the following experiments, due to efficiency considerations, we use the “tight” rule extraction in `cdec` that is more strict than the standard “loose” rule extraction, which generates a reduced rule set and, thus, a reduced reachability. We show the reachability distributions of both tight and loose rule extraction in Figure 4.

4.4 Evaluation on IWSLT

For IWSLT, we first compare the performance from various update methods in Figure 5. The max-violation method is more than 15 BLEU

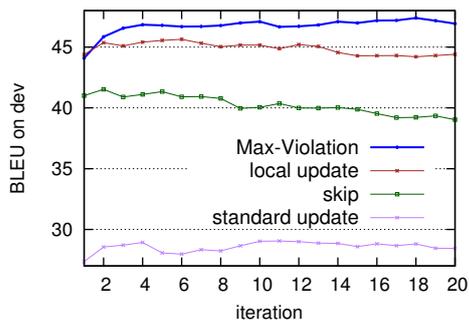


Figure 5: Comparison of various update methods.

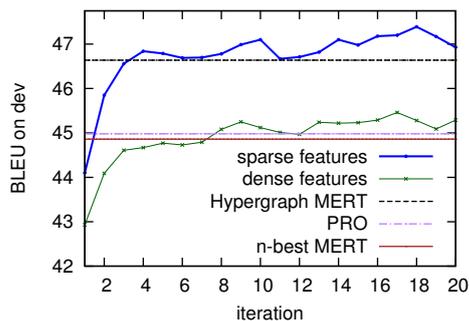


Figure 6: Sparse features (Word-Edges) contribute ~ 2 BLEU points, outperforming PRO and MERT.

points better than the standard perceptron (also known as “bold-update” in Liang et al. (2006)) which updates at the root of the derivation tree.^{3,4} This can be explained by the fact that in training $\sim 58\%$ of the standard updates are invalid (i.e., they do not fix any violation). We also use the “skip” strategy of Zhang et al. (2013) which updates at the root of the derivation only when it fixes a search error, avoiding all invalid updates. This achieves ~ 10 BLEU better than the standard update, but is still more than ~ 5 BLEU worse than Max-Violation update. Finally we also try the “local-update” method from Liang et al. (2006) which updates towards the derivation with the best Bleu^{+1} in the root group $S_{[0:|x|]}$. This method is about 2 BLEU points worse than max-violation.

We further investigate the contribution of sparse features in Figure 6. On the development set, max-violation update without Word-Edges features achieves BLEU similar to n -best MERT and

³We find that while MAXFORCE generates translations of length ratio close to 1 during training, the length ratios on dev/test sets are significantly lower, due to OOVs. So we run a binary search for the length penalty weight after each training iteration to tune the length ratio to ~ 0.97 on dev set.

⁴We report BLEU with **averaged** reference lengths.

algorithm	# feats	dev	test
n -best MERT	18	44.9	47.9
Hypergraph MERT	18	46.6	50.7
PRO	18	45.0	49.5
local update perc.	443K	45.6	49.1
MAXFORCE	529K	47.4	51.5

Table 2: BLEU scores (with 16 references) of various training algorithms on IWSLT09.

algorithm	# feats	dev	test
Hypergraph MERT	18	27.3	23.0
PRO	18	26.4	22.7
MAXFORCE	4.5M	27.7	23.9

Table 3: BLEU scores (with 4 references) of various training algorithms on FBIS.

PRO, but lower than Hypergraph MERT. Adding simple Word-Edges features improves BLEU by ~ 2 points, outperforming the very strong Hypergraph MERT baseline by ~ 1 point. See Table 2 for details. The results of n -best MERT, Hypergraph MERT, and PRO are averages from 3 runs.

4.5 Evaluation on FBIS

Table 3 shows BLEU scores of Hypergraph MERT, PRO, and MAXFORCE on FBIS. MAXFORCE activates 4.5M features, and achieves +1.2 BLEU over PRO and +0.9 BLEU over Hypergraph MERT. The training time (on 32 cores) for Hypergraph MERT and PRO is about 30 min. on the dev set, and is about 5 hours for MAXFORCE on the training set.

5 Conclusions

We have presented a latent-variable violation-fixing framework for general structured prediction problems with inexact search over hypergraphs. Its application on HIERO brings significant improvement in BLEU, compared to algorithms that are specially designed for MT tuning such as MERT and PRO.

Acknowledgment

Part of this work was done during K. Z.’s internship at IBM. We thank Martin Čmejrek and Lemao Liu for discussions, David Chiang for pointing us to `pycdec`, Dan Gildea for Footnote 2, and the anonymous reviewers for comments. This work is supported by DARPA FA8750-13-2-0041 (DEFT), DARPA HR0011-12-C-0015 (BOLT), and a Google Faculty Research Award.

References

- Abhishek Arun and Philipp Koehn. 2007. Online learning methods for discriminative training of phrase based statistical machine translation. *Proc. of MT Summit XI*, 2(5):29.
- Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In *ACL*, pages 200–208.
- Victor Chahuneau, Noah Smith, and Chris Dyer. 2012. pycdec: A python interface to cdec. *Prague Bulletin of Mathematical Linguistics*, (98).
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of ACL*, pages 173–180, Ann Arbor, Michigan, June.
- David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of EMNLP 2008*.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL*.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP*.
- Chris Dyer, Adam Lopez, Juri Ganitkevitch, Johnathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the ACL*.
- Jeffrey Flanigan, Chris Dyer, and Jaime Carbonell. 2013. Large-scale discriminative training for statistical machine translation using held-out line search. In *Proceedings of NAACL 2013*.
- Spence Green, Sida Wang, Daniel Cer, and Christopher D Manning. 2013. Fast and adaptive online training of feature-rich translation models. *to appear) ACL*.
- Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of EMNLP*.
- Liang Huang and David Chiang. 2007. Forest rescore: Fast decoding with integrated language models. In *Proceedings of ACL*, Prague, Czech Rep., June.
- Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of NAACL*.
- Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of the ACL: HLT*, Columbus, OH, June.
- P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of ACL*.
- Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of the Joint Conference of ACL and AFNLP*.
- Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of ACL*.
- Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of COLING-AACL*, Sydney, Australia, July.
- Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd ACL*.
- Franz Joseph Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*, pages 160–167.
- Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of EMNLP-CoNLL*.
- Heng Yu, Liang Huang, Haitao Mi, and Kai Zhao. 2013. Max-violation perceptron and forced decoding for scalable MT training. In *Proceedings of EMNLP*.
- Hao Zhang, Liang Huang, Kai Zhao, and Ryan McDonald. 2013. Online learning with inexact hypergraph search. In *Proceedings of EMNLP*.
- Kai Zhao and Liang Huang. 2013. Minibatch and parallelization for online large margin structured learning. In *Proceedings of NAACL 2013*.

Punctuation Processing for Projective Dependency Parsing*

Ji Ma[†], Yue Zhang[‡] and Jingbo Zhu^{†*}

[†]Northeastern University, Shenyang, China

[‡]Singapore University of Technology and Design, Singapore

*Hangzhou YaTuo Company, 358 Wener Rd., Hangzhou, China, 310012

majineu@gmail.com

yue_zhang@sutd.edu.sg

zhujingbo@mail.neu.edu.cn

Abstract

Modern statistical dependency parsers assign lexical heads to punctuations as well as words. Punctuation parsing errors lead to low parsing accuracy on words. In this work, we propose an alternative approach to addressing punctuation in dependency parsing. Rather than assigning lexical heads to punctuations, we treat punctuations as properties of their neighbouring words, used as features to guide the parser to build the dependency graph. Integrating our method with an arc-standard parser yields a 93.06% unlabelled attachment score, which is the best accuracy by a single-model transition-based parser reported so far.

1 Introduction

The task of dependency parsing is to identify the lexical head of each of the tokens in a string. Modern statistical parsers (McDonald et al., 2005; Nivre et al., 2007; Huang and Sagae, 2010; Zhang and Nivre, 2011) treat all the tokens equally, assigning lexical heads to punctuations as well as words. Punctuations arguably play an important role in syntactic analysis. However, there are a number of reasons that it is not necessary to parse punctuations:

First, the lexical heads of punctuations are not as well defined as those of words. Consequently, punctuations are not as consistently annotated in treebanks as words, making it harder to parse punctuations. For example, modern statistical parsers achieve above 90% unlabelled attachment score (UAS) on words. However, the UAS on punctuations are generally below 85%.

Moreover, experimental results showed that parsing accuracy of content words drops on sentences which contain higher ratios of punctuations. One reason for this result is that projective dependency parsers satisfy the “no crossing links” constraint, and errors in punctuations may prevent correct word-word dependencies from being created (see section 2). In addition, punctuations cause certain type of features inaccurate. Take valency features for example, previous work (Zhang and Nivre, 2011) has shown that such features are important to parsing accuracy, e.g., it may inform the parser that a verb already has two objects attached to it. However, such information might be inaccurate when the verb’s modifiers contain punctuations.

Ultimately, it is the dependencies between words that provide useful information for real world applications. Take machine translation or information extraction for example, most systems take advantage of the head-modifier relationships between word pairs rather than word-punctuation pairs to make better predictions. The fact that most previous work evaluates parsing accuracies without taking punctuations into account is also largely due to this reason.

Given the above reasons, we propose an alternative approach to punctuation processing for dependency parsing. In this method, punctuations are not associated with lexical heads, but are treated as properties of their neighbouring words.

Our method is simple and can be easily incorporated into state-of-the-art parsers. In this work, we report results on an arc-standard transition-based parser. Experiments show that our method achieves about 0.90% UAS improvement over the greedy baseline parser on the standard Penn Treebank test set. Although the improvement becomes smaller as the beam width grows larger, we still achieved 93.06% UAS with a beam of width 64, which is the best result for transition-based parsers

This work was done while the first author was visiting SUTD

Length	1 ~ 20			21 - 40			41 - 60		
	0 ~ 15	15 ~ 30	> 30	0 ~ 15	15 ~ 30	> 30	0 ~ 15	15 ~ 30	> 30
E-F	94.56	92.88	87.67	91.84	91.82	83.87	89.83	88.01	—
A-S	93.87	92.00	90.05	90.81	90.15	75.00	88.06	88.89	—
A-S-64	95.28	94.43	88.15	92.96	92.63	76.61	90.78	88.76	—
MST	94.90	93.55	88.15	92.45	93.11	77.42	90.89	89.77	—

Table 2: Parsing accuracies vs punctuation ratios, on the development set

System	E-F	A-S	A-S-64	MST
Dev UAS	91.83	90.71	93.02	92.56
Test UAS	91.75	90.34	92.84	92.10
Dev UAS-p	83.20	79.69	84.80	84.42
Test UAS-p	84.67	79.64	87.80	85.67
Dev ⁻ UAS	90.64	89.55	91.87	90.11
Test ⁻ UAS	90.40	89.33	91.75	89.82

Table 1: Parsing accuracies. “E-F” and “MST” denote easy-first parser and MSTparser, respectively. “A-S” and “A-S 64” denote our arc-standard parser with beam width 1 and 64, respectively. “UAS” and “UAS-p” denote word and punctuation unlabelled attachment score, respectively. “—” denotes the data set with punctuations removed.

reported so far. Our code will be available at <https://github.com/majineu/Parser/Punc/A-STD>.

2 Influence of Punctuations on Parsing

In this section, we conduct a set of experiments to show the influence of punctuations on dependency parsing accuracies.

2.1 Setup

We use the Wall Street Journal portion of the Penn Treebank with the standard splits: sections 02-21 are used as the training set; section 22 and section 23 are used as the development and test set, respectively. Penn2Malt is used to convert bracketed structures into dependencies. We use our own implementation of the Part-Of-Speech (POS) tagger proposed by Collins (2002) to tag the development and test sets. Training set POS tags are generated using 10-fold jack-knifing. Parsing accuracy is evaluated using unlabelled attachment score (UAS), which is the percentage of words that are assigned the correct lexical heads.

To show that the influence of punctuations on parsing is independent of specific parsing algorithms, we conduct experiments using three parsers, each representing a different parsing methodology: the open source MST-

Parser¹(McDonald and Pereira, 2006), our own re-implementation of an arc-standard transition-based parser (Nivre, 2008), which is trained using global learning and beam-search (Zhang and Clark, 2008) with a rich feature set (Zhang and Nivre, 2011)², and our own re-implementation of the easy-first parser (Goldberg and Elhadad, 2010) with an extended feature set (Ma et al., 2013).

2.2 Punctuations and Parsing Accuracy

Our first experiment is to show that, compared with words, punctuations are more difficult to parse and to learn. To see this, we evaluate the parsing accuracies of the selected parsers on words and punctuations, separately. Results are listed in Table 1, where row 2 and row 3 list the UAS of words (all excluding punctuations) on the development and test set, respectively. Row 4 and row 5 list accuracies of punctuations (all excluding words) on the development and test set, respectively. We can see that although all the parsers achieve above 90% UAS on words, the UAS on punctuations are mostly below 85%.

As for learning, we calculate the percentage of parameter updates that are caused by associating punctuations with incorrect heads during training of the easy-first parser³. The result is that more than 31% of the parameter updates are caused due to punctuations, though punctuations account for only 11.6% of the total tokens in the training set.

The fact that parsers achieve low accuracies on punctuations is to some degree expected, because the head of a punctuation mark is linguistically less well-defined. However, a related problem is

¹We trained a second order labelled parser with all the configurations set to the default value. The code is publicly available at <http://sourceforge.net/projects/mstparser/>

²Some feature templates in Zhang and Nivre (2011) involve head word and head POS tags which are not available for an arc-standard parser. Interestingly, without those features our arc-standard parser still achieves 92.84% UAS which is comparable to the 92.90% UAS obtained by the arc-eager parser of Zhang and Nivre (2011)

³For the greedy easy-first parser, whether a parameter update is caused by punctuation error can be determined with no ambiguity.

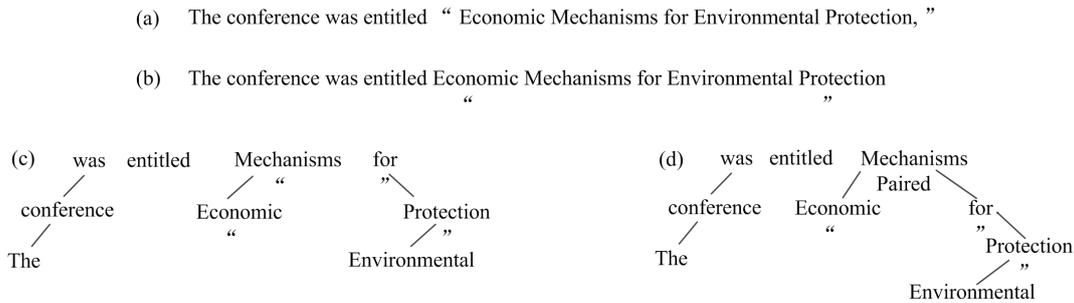


Figure 1: Illustration of processing paired punctuation. The property of a word is denoted by the punctuation below that word.

that parsing accuracy on *words* tends to drop on the sentences which contain high ratio of *punctuations*. To see this, we divide the sentences in the development set into sub-sets according the punctuation ratio (percentage of punctuations that a sentence contains), and then evaluate parsing accuracies on the sub-sets separately.

The results are listed in Table 2. Since long sentences are inherently more difficult to parse, to make a fair comparison, we further divide the development set according to sentence lengths as shown in the first row⁴. We can see that most of the cases, parsing accuracies drop on sentences with higher punctuation ratios. Note that this negative effect on parsing accuracy might be overlooked since most previous work evaluates parsing accuracy *without* taking punctuations into account.

By inspecting the parser outputs, we found that error propagation caused by assigning incorrect head to punctuations is one of the main reason that leads to this result. Take the sentence shown in Figure 1 (a) for example, the word *Mechanisms* is a modifier of *entitled* according to the gold reference. However, if the quotation mark, “, is incorrectly recognized as a modifier of *was*, due to the “no crossing links” constraint, the arc between *Mechanisms* and *entitled* can never be created.

A natural question is whether it is possible to reduce such error propagation by simply removing all punctuations from parsing. Our next experiment aims at answering this question. In this experiment, we first remove all punctuations from the original data and then modify the dependency arcs accordingly in order to maintain word-word dependencies in the original data. We re-train the parsers on the modified training set and evaluate

⁴1694 out of 1700 sentences on the development set with length no larger than 60 tokens

parsing accuracies on the modified data.

Results are listed in row 6 and row 7 of Table 1. We can see that parsing accuracies on the modified data drop significantly compared with that on the original data. The result indicates that by removing punctuations, we lose some information that is important for dependency parsing.

3 Punctuation as Properties

In our method, punctuations are treated as properties of its neighbouring words. Such properties are used as additional features to guide the parser to construct the dependency graph.

3.1 Paired Punctuation

Our method distinguishes *paired* punctuations from other punctuations. Here paired punctuations include brackets and quotations marks, whose Penn Treebank POS tags are the following four:

-LRB- -RRB- “ ”

The characteristics of paired punctuations include: (1) they typically exist in pairs; (2) they serve as *boundaries* that there is only one dependency arc between the words inside the boundaries and the words outside. Take the sentence in Figure 1 (a) for example, the only arc cross the boundary is (Mechanisms, entitled) where entitled is the head.

To utilize such boundary information, we further classify paired punctuations into two categories: those that serve as the beginning of the boundary, whose POS tags are either -LRB- or “, denoted by BPUNC; and those that serve as the end of the boundary, denoted by EPUNC.

Before parsing starts, a preprocessing step is used to first attach the paired punctuations as properties of their neighbouring words, and then remove them from the sentence. In particular,

unigram	for p in $\beta_0, \beta_1, \beta_2, \beta_3, \sigma_0, \sigma_1, \sigma_2$	p_{punc}
	for p in $\beta_0, \beta_1, \beta_2, \sigma_0, \sigma_1$	$p_{punc} \odot p_w, p_{punc} \odot p_t$
bigram	for p, q in $(\sigma_0, \beta_0), (\sigma_0, \beta_1), (\sigma_0, \beta_2), (\sigma_0, \sigma_1), (\sigma_0, \sigma_2)$	$p_{punc} \odot q_{punc}, p_{punc} \odot q_t, p_{punc} \odot q_w$
	for p, q in $(\sigma_2, \sigma_0), (\sigma_1, \sigma_0), (\sigma_2, \sigma_0)$	$p_{punc} \odot q_t, p_{punc} \odot p_t \odot q_t$
	for p, q in $(\sigma_2, \sigma_0), (\sigma_1, \sigma_0), (\sigma_0, \beta_0)$	$d_{pq} \odot p_{punc} \odot p_t \odot q_t$

Table 3: Feature templates. For an element p either on σ or β of an arc-standard parser, we use p_{punc} , p_w and p_t to denote the punctuation property, head word and head tag of p , respectively. d_{pq} denotes the distance between the two elements p and q .

we attach BPUNCs to their right neighbours and EPUNCs to their left neighbours, as shown in Figure 1 (b). Note that in Figure 1 (a), the left neighbour of ” is also a punctuation. In such cases, we simply remove these punctuations since the existence of paired punctuations already indicates that there should be a boundary.

During parsing, when a dependency arc with lexical head w_h is created, the property of w_h is updated by the property of its left (or right) most child to keep track whether there is a BPUNC (or EPUNC) to the left (or right) side of the sub-tree rooted at w_h , as shown in Figure 1 (c). When BPUNCs and EPUNCs meet each other at w_h , a PAIRED property is assigned to w_h to capture that the words within the paired punctuations form a sub-tree, rooted at w_h . See Figure 1 (d).

3.2 Practical Issues

It is not uncommon that two BPUNCs appear adjacent to each other. For example,

(“*Congress’s Environmental Buccaneers,*” *Sept. 18*).

In our implementation, BPUNC or EPUNC properties are implemented using flags. In the example, we set two flags “ and (on the word Congress’s. When BPUNC and EPUNC meet each other, the corresponding flags are turned off. In the example, when Congress’s is identified as a modifier of Buccaneers, the ” flag of Buccaneers is turned off. However, we do not assign a PAIRED property to Buccaneers since its (flag is still on. The PAIRED property is assigned only when all the flags are turned off.

3.3 Non-Paired Punctuations

Though some types of non-paired punctuations may capture certain syntactic patterns, we do not make further distinctions between them, and treat these punctuations uniformly for simplicity.

Before parsing starts and after the preprocessing step for paired punctuations, our method employs

a second preprocessing step to attach non-paired punctuations to their left neighbouring words. It is guaranteed that the property of the left neighbouring words of non-paired punctuations must be empty. Otherwise, it means the non-paired punctuation is adjacent to a paired punctuation. In such cases, the non-paired punctuation would be removed in the first processing step.

During parsing, non-paired punctuations are also passed bottom-up: the property of w_h is updated by its *right*-most dependent to keep track whether there is a punctuation to the right side of the tree rooted at w_h . The only special case is that if w_h already contains a BPUNC property, then our method simply ignores the non-paired property since we maintain the boundary information with the highest priority.

3.4 Features

We incorporate our method into the arc-standard transition-based parser, which uses a stack σ to maintain partially constructed trees and a buffer β for the incoming words (Nivre, 2008). We design a set of features to exploit the potential of using punctuation properties for the arc-standard parser.

The feature templates are listed in Table 3. In addition to the features designed for paired punctuations, such as bigram punctuation features listed in line 3 of Table 3, we also design features for non-paired punctuations. For example, the distance features in line 5 of Table 3 is used to capture the pattern that if a word w with comma property is the left modifier of a noun or a verb, the distance between w and its lexical head is often larger than 1. In other words, they are not adjacent.

4 Results

Our first experiment is to investigate the effect of processing paired punctuations on parsing accuracy. In this experiment, the method introduced in Section 3.1 is used to process paired punctuations, and the non-paired punctuations are left un-

s	Baseline	Paired	All
1	90.76	91.25	91.47
2	91.88	92.06	92.34
4	92.50	92.61	92.70
8	92.73	92.76	92.82
16	92.90	92.94	92.99
64	92.99	93.04	93.10

Table 4: Parsing accuracies on the development set. s denotes the beam width.

touched. Feature templates used in this experiment are those listed in the top three rows of Table 3 together with those used for the baseline arc-standard parser.

Results on the development set are shown in the second column of Table 4. We can see that when the beam width is set to 1, our method achieves an 0.49 UAS improvement. By comparing the outputs of the two parsers, two types of errors made by the baseline parser are effectively corrected.

The first is that our method is able to capture the pattern that there is only one dependency arc between the words within the paired-punctuations and the words outside, while the baseline parser sometimes creates more dependency arcs that cross the boundary.

The second is more interesting. Our method is able to capture that the root, w_h , of the sub-tree within the paired-punctuation, such as “Mechanisms” in Figure 1, generally serves as a modifier of the words outside, while the baseline parser occasionally make w_h as the head of the sentence.

As we increase the beam width, the improvement of our method over the baseline becomes smaller. This is as expected, since beam search also has the effect of reducing error propagation (Zhang and Nivre, 2012), thereby alleviating the errors caused by punctuations.

In the last experiment, we examine the effect of incorporating all punctuations using the method introduced in Section 2. In this experiment, we use all the feature templates in Table 3 and those in the baseline parser. Results are listed in the fourth column of Table 4, which shows that parsing accuracies can be further improved by also processing non-paired punctuations. The overall accuracy improvement when the beam width is 1 reaches 0.91%. The extra improvements mainly come from better accuracies on the sentences with comma. However, the exact type of errors that are corrected by using non-paired punctuations is more difficult to summarize.

system	UAS	Comp	Root
Baseline	90.38	37.71	89.45
All-Punc	91.32	41.35	92.43
Baseline-64	92.84	46.90	95.57
All-Punc-64	93.06	48.55	95.53
Huang 10	92.10	—	—
Zhang 11	92.90	48.00	91.80
Choi 13	92.96	—	—
Bohnet 12	93.03	—	—

Table 5: Final result on the test set.

The final results on the test set are listed in Table 5⁵. Table 5 also lists the accuracies of state-of-the-art transition-based parsers. In particular, “Huang 10” and “Zhang 11” denote Huang and Sagae (2010) and Zhang and Nivre (2011), respectively. “Bohnet 12” and “Choi 13” denote Bohnet and Nivre (2012) and Choi and Mccallum (2013), respectively. We can see that our method achieves the best accuracy for single-model transition-based parsers.

5 Conclusion and Related Work

In this work, we proposed to treat punctuations as properties of context words for dependency parsing. Experiments with an arc-standard parser showed that our method effectively improves parsing performance and we achieved the best accuracy for single-model transition-based parser.

Regarding punctuation processing for dependency parsing, Li et al. (2010) proposed to utilize punctuations to segment sentences into small fragments and then parse the fragments separately. A similar approach is proposed by Spitzkovsky et al. (2011) which also designed a set of constraints on the fragments to improve unsupervised dependency parsing.

Acknowledgements

We highly appreciate the anonymous reviewers for their insightful suggestions. This research was supported by the National Science Foundation of China (61272376; 61300097; 61100089), the Fundamental Research Funds for the Central Universities (N110404012), the research grant T2MOE1301 from Singapore Ministry of Education (MOE) and the start-up grant SRG ISTD2012038 from SUTD.

⁵The number of training iteration is determined using the development set.

References

- Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1455–1465, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Jinho D. Choi and Andrew McCallum. 2013. Transition-based dependency parsing with selectional branching. In *In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*, EMNLP '02, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Yoav Goldberg and Michael Elhadad. 2010. An efficient algorithm for easy-first non-directional dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 742–750, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In Jan Hajic, Sandra Carberry, and Stephen Clark, editors, *ACL*, pages 1077–1086. The Association for Computer Linguistics.
- Zhenghua Li, Wanxiang Che, and Ting Liu. 2010. Improving dependency parsing using punctuation. In Minghui Dong, Guodong Zhou, Haoliang Qi, and Min Zhang, editors, *IJALP*, pages 53–56. IEEE Computer Society.
- Ji Ma, Jingbo Zhu, Tong Xiao, and Nan Yang. 2013. Easy-first pos tagging and dependency parsing with beam search. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 110–114, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-2006)*, volume 6, pages 81–88.
- Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 91–98, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.
- Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.
- Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2011. Punctuation: Making a point in unsupervised dependency parsing. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL-2011)*.
- Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Yue Zhang and Joakim Nivre. 2012. Analyzing the effect of global learning and beam-search on transition-based dependency parsing. In *Proceedings of COLING 2012: Posters*, pages 1391–1400, Mumbai, India, December. The COLING 2012 Organizing Committee.

Transforming trees into hedges and parsing with “hedgebank” grammars

Mahsa Yarmohammadi[†], Aaron Dunlop[†] and Brian Roark[°]

[†]Oregon Health & Science University, Portland, Oregon [°]Google, Inc., New York
yarmoham@ohsu.edu, {aaron.dunlop, roarkbr}@gmail.com

Abstract

Finite-state chunking and tagging methods are very fast for annotating non-hierarchical syntactic information, and are often applied in applications that do not require full syntactic analyses. Scenarios such as incremental machine translation may benefit from some degree of hierarchical syntactic analysis without requiring fully connected parses. We introduce *hedge parsing* as an approach to recovering constituents of length up to some maximum span L . This approach improves efficiency by bounding constituent size, and allows for efficient segmentation strategies prior to parsing. Unlike shallow parsing methods, hedge parsing yields internal hierarchical structure of phrases within its span bound. We present the approach and some initial experiments on different inference strategies.

1 Introduction

Parsing full hierarchical syntactic structures is costly, and some NLP applications that could benefit from parses instead substitute shallow proxies such as NP chunks. Models to derive such non-hierarchical annotations are finite-state, so inference is very fast. Still, these partial annotations omit all but the most basic syntactic segmentation, ignoring the abundant local structure that could be of utility even in the absence of fully connected structures. For example, in incremental (simultaneous) machine translation (Yarmohammadi et al., 2013), sub-sentential segments are translated independently and sequentially, hence the fully-connected syntactic structure is not generally available. Even so, locally-connected source language parse structures can inform both segmentation and translation of each segment in such a translation scenario.

One way to provide local hierarchical syntactic

structures without fully connected trees is to focus on providing full hierarchical annotations for structures within a local window, ignoring global constituents outside that window. We follow the XML community in naming structures of this type *hedges* (not to be confused with the rhetorical device of the same name), due to the fact that they are like smaller versions of trees which occur in sequences. Such structures may be of utility to various structured inference tasks, as well as within a full parsing pipeline, to quickly constrain subsequent inference, much as finite-state models such as supertagging (Bangalore and Joshi, 1999) or chart cell constraints (Roark and Hollingshead, 2008; Roark et al., 2012) are used.

In this paper, we consider the problem of *hedge parsing*, i.e., discovering every constituent of length up to some span L . Similar constraints have been used in dependency parsing (Eisner and Smith, 2005; Dreyer et al., 2006), where the use of hard constraints on the distance between heads and dependents is known as vine parsing. It is also reminiscent of so-called Semi-Markov models (Sarawagi and Cohen, 2004), which allow finite-state models to reason about segments rather than just tags by imposing segment length limits. In the XML community, trees and hedges are used for models of XML document instances and for the contents of elements (Brüggemann-Klein and Wood, 2004). As far as we know, this paper is the first to consider this sort of partial parsing approach for natural language.

We pursue this topic via tree transformation, whereby non-root non-terminals labeling constituents of span $> L$ in the tree are recursively elided and their children promoted to attach to their parent. In such a way, hedges are sequentially connected to the top-most non-terminal in the tree, as demonstrated in Figure 1. After applying such a transform to a treebank, we can induce grammars and modify parsing to search as needed to recover just these constituents.

In this paper, we propose several methods to

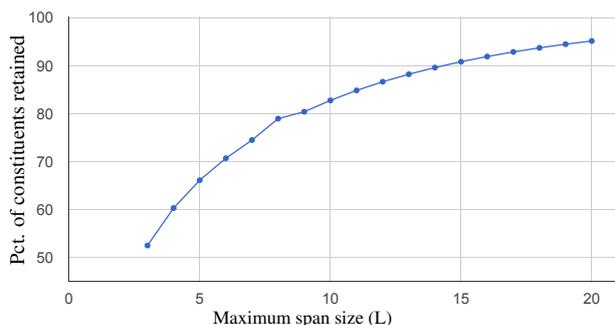


Figure 2: Percentage of constituents retained at various span length parameters

hedge-parsing accuracy, as it has access to rich contextual information (as compared to grammars trained on transformed trees). Naturally, inference will be slow; we aim to improve efficiency upon this baseline while minimizing accuracy loss.

Since we limit the span of non-terminal labels, we can constrain the search performed by the parser, greatly reduce the CYK processing time. In essence, we perform no work in chart cells spanning more than L words, except for the cells along the periphery of the chart, which are just used to connect the hedges to the root. Consider the flat tree in Figure 1(b). For use by a CYK parsing algorithm, trees are binarized prior to grammar induction, resulting in special non-terminals created by binarization. Other than the symbol at the root of the tree, the only constituents with span length greater than L in the binarized tree will be labeled with these special binarization non-terminals. Further, if the binarization systematically groups the leftmost or the rightmost children under these new non-terminals (the most common strategy), then constituents with span greater than L will either begin at the first word (leftmost grouping) or end at the last word (rightmost), further constraining the number of cells in the chart requiring work.

Complexity of parsing with a full CYK parser is $O(n^3|G|)$ where n is the length of input and $|G|$ is the grammar size constant. In contrast, complexity of parsing with a hedge constrained CYK is reduced to $O((nL^2 + n^2)|G|)$. To see that this is the case, consider that there are $O(nL)$ cells of span L or less, and each has a maximum of L midpoints, which accounts for the first term. Beyond these, there are $O(n)$ remaining active cells with $O(n)$ possible midpoints, which accounts for the second term. Note also that these latter cells (spanning $> L$ words) may be less expensive, as the set of possible non-terminals is reduced to only those introduced by binarization.

It is possible to parse with a standardly induced

PCFG using this sort of hedge constrained parsing that only considers a subset of the chart cells, and speedups are achieved, however this is clearly non-optimal, since the model is ill-suited to combining hedges into flat structures at the root of the tree. Space constraints preclude inclusion of trials with this method, but the net result is a severe degradation in accuracy (tens of points of F-measure) versus standard parsing. Thus, we train a grammar in a matched condition, which we call it a *hedgebank grammar*. A hedgebank grammar is a fully functional PCFG which is learned from a hedge transformed treebank. A hedgebank grammar can be used with any standard parsing algorithm, i.e., these are not generally finite-state equivalent models. However, using the Berkeley grammar learner (see §3), we find that hedgebank grammars are typically smaller than treebank grammars, reducing the grammar constant and contributing to faster inference.

A unique property of hedge constituents compared to constituents in the original parse trees is that they are sequentially connected to the topmost node. This property enables us to chunk the sentence into segments that correspond to complete hedges, and parse the segments independently (and simultaneously) instead of parsing the entire sentence. In section 2.3, we present our approach to hedge segmentation.

In all scenarios where the chart is constrained to search for hedges, we learn a hedgebank grammar, which is matched to the maximum length allowed by the parser. In the pre-segmentation scenario, we first decompose the hedge transformed treebank into its hedge segments and then learn a hedgebank grammar from the new corpus.

2.3 Hedge Segmentation

In this section we present our segmentation model which takes the input sentence and chunks it into appropriate segments for hedge parsing. We treat this as a binary classification task which decides if a word can begin a new hedge. We use hedge segmentation as a finite-state pre-processing step for hedge context-free parsing.

Our task is to learn which words can begin (B) a hedge constituent. Given a set of labeled pairs (S, H) where S is a sentence of n words $w_1 \dots w_n$ and H is its hedge parse tree, word w_b belongs to B if there is a hedge constituent spanning $w_b \dots w_e$ for some $e \geq b$ and w_b belongs to \bar{B} otherwise. To predict the hedge boundaries more accurately, we grouped consecutive unary or POS-

tag hedges together under a new non-terminal labeled G . Unlabeled segmentation tags for the words in the example sentence in Figure 1(b) are:

“Analysts/ B are/ \bar{B} concerned/ \bar{B} that/ \bar{B} much/ B
of/ \bar{B} the/ \bar{B} high-yield/ \bar{B} market/ \bar{B} will/ B
remain/ \bar{B} treacherous/ \bar{B} for/ \bar{B} investors/ \bar{B} . B ”

In addition to the simple unlabeled segmentation with B and \bar{B} tags, we try a labeled segmentation with B_C and \bar{B}_C tags where C is hedge constituent type. We restrict the types to the most important types – following the 11 chunk types annotated in the CoNLL-2000 chunking task (Sang and Buchholz, 2000) – by replacing all other types with a new type OUT . Thus, “Analysts” is labeled B_G ; “much”, B_{NP} ; “will”, B_{VP} and so on.

To automatically predict the class of each word position, we train a multi-class classifier from labeled training data using a discriminative linear model, learning the model parameters with the averaged perceptron algorithm (Collins, 2002). We follow Roark et al. (2012) in the features they used to label words as beginning or ending constituents. The segmenter extracts features from word and POS-tag input sequences and hedge-boundary tag output sequences. The feature set includes trigrams of surrounding words, trigrams of surrounding POS tags, and hedge-boundary tags of the previous words. An additional orthographical feature set is used to tag rare¹ and unknown words. This feature set includes prefixes and suffixes of the words (up to 4 characters), and presence of a hyphen, digit, or an upper-case character. Reported results are for a Markov order-2 segmenter, which includes features with the output classes of the previous two words.

3 Experimental Results

We ran all experiments on the WSJ Penn Treebank corpus (Marcus et al., 1999) using section 2-21 for training, section 24 for development, and section 23 for testing. We performed exhaustive CYK parsing using the BUBS parser² (Bodenstab et al., 2011) with Berkeley SM6 latent-variable grammars (Petrov and Klein, 2007) learned by the Berkeley grammar trainer with default settings. We compute accuracy from the 1-best Viterbi tree extracted from the chart using the standard EVALB script. Accuracy results are reported as precision, recall and F1-score, the harmonic mean between the two. In all trials, we evaluate accuracy with respect to the hedge transformed reference

¹Rare words occur less than 5 times in the training data.

²<https://code.google.com/p/bubs-parser>

Parser	Hedge Parsing Acc/Eff			
	P	R	F1	w/s
Full w/full CYK	88.8	89.2	89.0	2.4
Hedgebank	87.6	84.4	86.0	25.7

Table 1: Hedge parsing results on section 24 for $L = 7$.

treebank, i.e., we are not penalizing the parser for not discovering constituents longer than the maximum length. Segmentation accuracy is reported as an F1-score of unlabeled segment bracketing. We ran timing tests on an Intel 2.66GHz processor with 3MB of cache and 2GB of memory. Note that segmentation time is negligible compared to the parsing time, hence is omitted in reported time. Efficiency results are reported as number of words parsed per second (w/s).

Table 1 presents hedge parsing accuracy on the development set for the full parsing baseline, where the output of regular PCFG parsing is transformed to hedges and evaluated, versus parsing with a hedgebank grammar, with no segmentation of the strings. We find an order of magnitude speedup of parsing, but at the cost of 3 percent F-measure absolute. Note that most of that loss is in recall, indicating that hedges predicted in that condition are nearly as reliable as in full parsing.

Table 2 shows the results on the development set when segmenting prior to hedge parsing. The first row shows the result with no segmentation, the same as the last row in Table 1 for ease of reference. The next row shows behavior with perfect segmentation. The final two rows show performance with automatic segmentation, using a model that includes either unlabeled or labeled segmentation tags, as described in the last section. Segmentation accuracy is better for the model with labels, although overall that accuracy is rather low. We achieve nearly another order of magnitude speedup over hedge parsing without segmentation, but again at the cost of nearly 5 percent F1.

Table 3 presents results of our best configurations on the eval set, section 23. The results show the same patterns as on the development set. Finally, Figure 3 shows the speed of inference, la-

Table 2: Hedge segmentation and parsing results on section 24 for $L = 7$.

Segmentation	Seg F1	Hedge Parsing Acc/Eff			
		P	R	F1	w/s
None	n/a	87.6	84.4	86.0	25.7
Oracle	100	91.3	88.9	90.1	188.6
Unlabeled	80.6	77.2	75.3	76.2	159.1
Labeled	83.8	83.1	79.5	81.3	195.8

Segmentation	Grammar	Segmentation Acc			Hedge Parsing Acc/Eff			
		P	R	F1	P	R	F1	w/s
None	Full w/full CYK			n/a	90.3	90.3	90.3	2.7
None	Hedgebank			n/a	88.3	85.3	86.8	26.2
Labeled	Hedgebank	84.0	86.6	85.3	85.1	81.1	83.0	203.0

Table 3: Hedge segmentation and parsing results on test data, section 23, for $L = 7$.

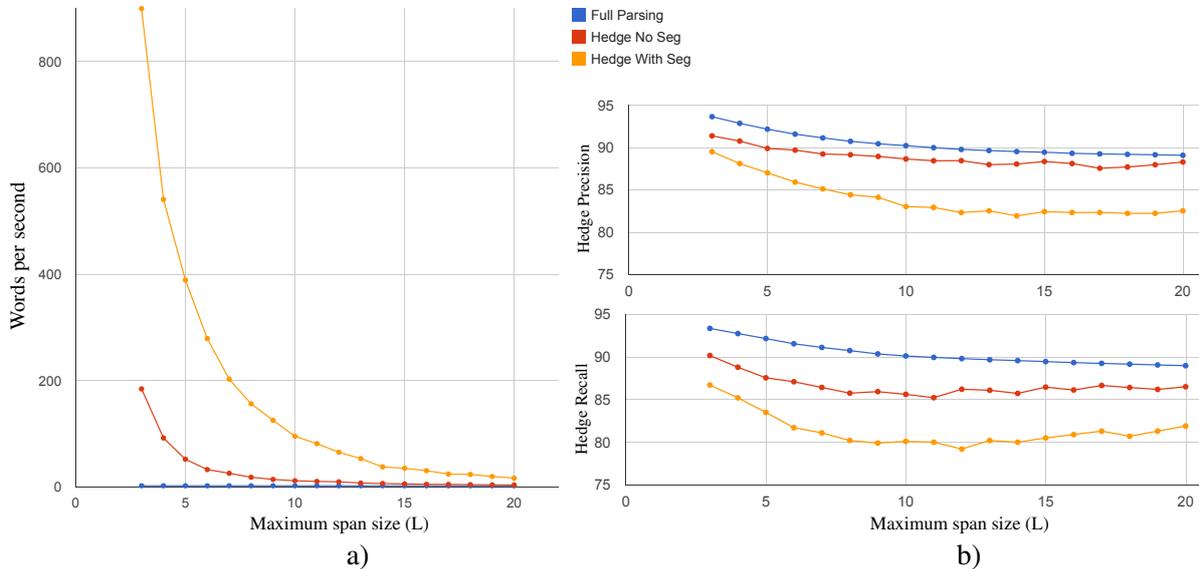


Figure 3: Hedge parsing a) efficiency, and b) accuracy on test data, section 23, for $L = 3-20$.

beled precision and labeled recall of annotating hedge constituents on the test set as a function of the maximum span parameter L , versus the baseline parser. Keep in mind that the number of reference constituents increases as L increases, hence both precision and recall can decrease as the parameter grows. Segmentation achieves large speedups for smaller L values, but the accuracy degradation is consistent, pointing to the need for improved segmentation.

4 Conclusion and Future Work

We proposed a novel partial parsing approach for applications that require a fast syntactic analysis of the input beyond shallow bracketing. The span-limit parameter allows tuning the annotation of internal structure as appropriate for the application domain, trading off annotation complexity against inference time. These properties make hedge parsing potentially very useful for incremental text or speech processing, such as streaming text analysis or simultaneous translation.

One interesting characteristic of these annotations is that they allow for string segmentation prior to inference, provided that the segment boundaries do not cross any hedge boundaries. We found that baseline segmentation models did pro-

vide a significant speedup in parsing, but that cascading errors remain a problem.

There are many directions of future work to pursue here. First, the current results are all for exhaustive CYK parsing, and we plan to perform a detailed investigation of the performance of hedgebank parsing with prioritization and pruning methods of the sort available in BUBS (Bodenstab et al., 2011). Further, this sort of annotation seems well suited to incremental parsing with beam search, which has been shown to achieve high accuracies even for fully connected parsing (Zhang and Clark, 2011). Improvements to the transform (e.g., grouping items not in hedges under non-terminals) and to the segmentation model (e.g., increasing precision at the expense of recall) could improve accuracy without greatly reducing efficiency. Finally, we intend to perform an extrinsic evaluation of this parsing in an on-line task such as simultaneous translation.

Acknowledgments

This work was supported in part by NSF grant #IIS-0964102. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the NSF.

References

- Srinivas Bangalore and Aravind K. Joshi. 1999. Supertagging: An approach to almost parsing. *Computational Linguistics*, 25(2):237–265.
- Nathan Bodenstab, Aaron Dunlop, Keith Hall, and Brian Roark. 2011. Beam-width prediction for efficient context-free parsing. In *Proceedings of the 49th Annual Meeting ACL: HLT*, pages 440–449.
- Anne Brüggemann-Klein and Derick Wood. 2004. Balanced context-free grammars, hedge grammars and pushdown caterpillar automata. In *Extreme Markup Languages*.
- Michael Collins. 2002. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proceedings of the conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–8.
- Markus Dreyer, David A. Smith, and Noah A. Smith. 2006. Vine parsing and minimum risk reranking for speed and precision. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL)*, pages 201–205.
- Jason Eisner and Noah A. Smith. 2005. Parsing with soft and hard constraints on dependency length. In *Proceedings of the Ninth International Workshop on Parsing Technology (IWPT)*, pages 30–41.
- Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. Treebank-3. Linguistic Data Consortium, Philadelphia.
- Slav Petrov and Dan Klein. 2007. Learning and inference for hierarchically split PCFGs. In *Proceedings of the 22nd national conference on Artificial intelligence*, pages 1663–1666.
- Brian Roark and Kristy Hollingshead. 2008. Classifying chart cells for quadratic complexity context-free inference. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 745–751.
- Brian Roark, Kristy Hollingshead, and Nathan Bodenstab. 2012. Finite-state chart constraints for reduced complexity context-free parsing pipelines. *Computational Linguistics*, 38(4):719–753.
- Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of Conference on Computational Natural Language Learning (CoNLL)*, pages 127–132.
- Sunita Sarawagi and William W. Cohen. 2004. Semi-Markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1185–1192.
- Mahsa Yarmohammadi, Vivek K. Rangarajan Sridhar, Srinivas Bangalore, and Baskaran Sankaran. 2013. Incremental segmentation and decoding strategies for simultaneous translation. In *Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP)*, pages 1032–1036.
- Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1):105–151.

Incremental Predictive Parsing with TurboParser

Arne Köhn and Wolfgang Menzel

Fachbereich Informatik
Universität Hamburg
{koehn, menzel}@informatik.uni-hamburg.de

Abstract

Most approaches to incremental parsing either incur a degradation of accuracy or they have to postpone decisions, yielding underspecified intermediate output. We present an incremental predictive dependency parser that is fast, accurate, and largely language independent. By extending a state-of-the-art dependency parser, connected analyses for sentence prefixes are obtained, which even predict properties and the structural embedding of upcoming words. In contrast to other approaches, accuracy for complete sentence analyses does not decrease.

1 Introduction

When humans communicate by means of a natural language, utterances are not produced at once but evolve over time. Human interaction benefits from this property by processing yet unfinished utterances and reacting on them. Computational parsing on the other hand is mostly performed on complete sentences, a processing mode which renders a responsive interaction based on incomplete utterances impossible.

When spoken language is analyzed, a mismatch between speech recognition and parsing occurs: If parsing does not work incrementally, the overall system loses all the desirable properties made possible by incremental processing. For speech dialogue systems, this leads to increased reaction times and an unnatural ping-pong style of interaction (Schlangen and Skantze, 2011).

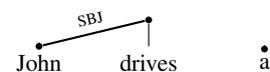
1.1 Desirable features of incremental parsers

Dependency parsing assigns a head and a dependency label to each word form of an input sentence and the resulting analysis of the sentence is usually required to form a tree. An incremental dependency

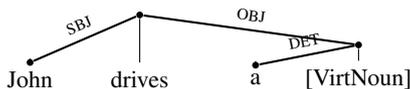
parser processes a sentence word by word, building analyses for sentence prefixes (*partial dependency analyses, PDA*), which are extended and modified in a piecemeal fashion as more words become available.

A PDA should come with three important (but partly contradictory) properties: beyond being accurate, it should also be as stable and informative as possible. Stability can be measured as the amount of structure (attachments and their labels) of a PDA a_i which is also part of the analysis a_n of the whole sentence. To be maximally informative, at least all available word forms should be integrated into the prefix PDA. Even such a simple requirement cannot easily be met without predicting a structural skeleton for the word forms in the upcoming part of the sentence (bottom-up prediction). Other predictions merely serve to satisfy completeness conditions (i.e. valency requirements) in an anticipatory way (top-down predictions). In fact, humans are able to derive such predictions and they do so during sentence comprehension (Sturt and Lombardo, 2005).

Without prediction, the sentence prefix “John drives a” of “John drives a car” can only be parsed as a disconnected structure:



The determiner remains unconnected to the rest of the sentence, because a possible head is not yet available. However, the determiner could be integrated into the PDA if the connection is established by means of a predicted word form, which has not yet been observed. Beuck et al. (2011) propose to use *virtual nodes* (VNs) for this purpose. Each VN represents exactly one upcoming word. Its lexical instantiation and its exact position remain unspecified. Using a VN, the prefix “John drives a” could then be parsed as follows, creating a fully connected analysis, which also satisfies the valency requirements of the finite verb.



This analysis is clearly more informative but still restricted to the existence of a noun filling the object role of "drives" without predicting its position. Although a VN does not specify the lexical identity of the word form it represents, it can nonetheless carry some information such as a coarse-grained part-of-speech category.

1.2 Related work

Parsers that produce incremental output are relatively rare: PLTag (Demberg-Winterfors, 2010) aims at psycholinguistic plausibility. It makes trade-offs in the field of accuracy and coverage (they report 6.2 percent of unparseable sentences on sentences of the Penn Treebank with less than 40 words). Due to its use of beam search, the incremental results are non-monotonic. Hassan et al. (2009) present a CCG-based parser that can parse in an incremental mode. The parser guarantees that every parse of an increment extends the previous parse monotonically. However, using the incremental mode without look-ahead, parsing accuracy drops from 86.70% to 59.01%. Obviously, insisting on strict monotonicity ($a_i \subseteq a_n$) is too strong a requirement, since it forces the parser to keep attachments that later turn out to be clearly wrong in light of new evidence.

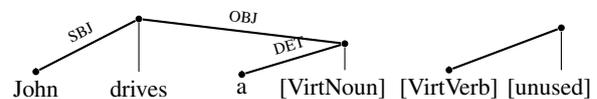
Being a transition-based parser, Maltparser (Nivre et al., 2007) does incremental parsing by design. It is, however, not able to predict upcoming structure and therefore its incremental output is usually fragmented into several trees. In addition, Maltparser needs a sufficiently large look-ahead to achieve high accuracy (Beuck et al., 2011).

Beuck et al. (2011) introduced incremental and predictive parsing using Weighted Constraint Dependency Grammar. While their approach does not decrease in incremental mode, it is much slower than most other parsers. Another disadvantage is its hand-written grammar which prevents the parser from being adapted to additional languages by simply training it on an annotated corpus and which makes it difficult to derive empirically valid conclusions from the experimental results.

2 Challenges for predictive parsing

Extending a dependency parser to incremental parsing with VNs introduces a significant shift in the problem to be solved: While originally the problem

was *where* to attach each word to (1), in the incremental case the additional problem arises, *which* VNs to include into the analysis (2). Problem (2), however, depends on the syntactic structure of the sentence prefix. Therefore, it is not possible to determine the VNs *before* parsing commences, but the decision has to be made *while* parsing is going on. We can resolve this issue by transforming problem (2) into problem (1) by providing the parser with an additional node, named *unused*. It is always attached to the special node 0 (the root node of every analysis) and it can only dominate VNs. *unused* and every VN it dominates are not considered part of the analysis. Using this idea, the problem of whether a VN should be included into the analysis is now reduced to the problem of where to attach that VN:



To enable the parser to include VNs into PDAs, a set of VNs has to be provided. While this set could include any number of VNs, we only include a set that covers most cases of prediction since rare virtual nodes have a very low a-priori probability of being included and additional VNs make the parsing problem more complex. This set is language-dependent and has to be determined in advance. It can be obtained by generating PDAs from a treebank and counting the occurrences of VNs in them. Eventually, a set of VNs is used that is a super-set of a large enough percentage (> 90%) of the observed sets.

3 Gold annotations for sentence prefixes

Annotating sentence prefixes by hand is prohibitively costly because the number of increments is a multitude of the number of sentences in the corpus. Beuck and Menzel (2013) propose an approach to automatically generate predictive dependency analyses from the annotation of full sentences. Their method tries to generate upper bounds for predictability which are relatively tight. Therefore, not everything that is deemed predictable by the algorithm is predictable in reality, but everything that is predictable should be deemed as predictable: Let W be all tokens of the sentence and P the set of tokens that lie in the prefix for which an incremental analysis should be generated. A word $w \in W \setminus P$ is assumed to be predictable ($w \in Pr$) if one of the following three criteria is met:

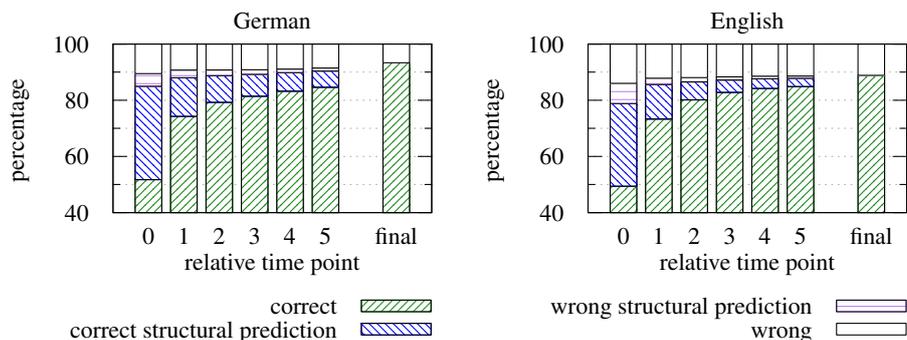
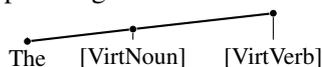


Figure 1: Results for TurboParser for German and English with gold standard PoS (labeled)

bottom-up prediction w lies on the path from some $w' \in P$ to 0. E. g., given the sentence prefix “The”, an upcoming noun and a verb is predicted:



top down prediction $\pi(w)$, the head of w , is in $P \cup Pr$, and w fills a syntactic role – encoded by its dependency label – that is *structurally determined*. That means w can be predicted independently of the lexical identity of $\pi(w)$. An example for this is the subject label: If $\pi(w)$ is in Pr and w is its subject, w is assumed to be predictable.

lexical top-down prediction $\pi(w) \in P$ and w fills a syntactic role that is determined by an already observed lexical item, e.g. the object role: If $\pi(w)$ is a known verb and w is its object, $w \in Pr$ because it is required by a valency of the verb.

While this procedure is language-independent, some language-specific transformations must be applied nonetheless. For English, parts of gapping coordinations can be predicted whereas others can not. For German, the transformations described in (Beuck and Menzel, 2013) have been used without further changes. Both sets of structurally and lexically determined roles are language dependent. The label sets for German have been adopted from (Beuck and Menzel, 2013), while the sets for English have been obtained by manually analyzing the PTB (Marcus et al., 1994) for predictability.

For words marked as predictable their existence and word class, but not their lexicalization and position can be predicted. Therefore, we replace the lexical item with “[virtual]” and generalize the part-of-speech tag to a more coarse grained one.

4 Predictive parsing with TurboParser

We adapt TurboParser (Martins et al., 2013) for incremental parsing because it does not impose structural constraints such as single-headedness in its core algorithm. For each parsing problem, it

creates an integer linear program – in the form of a factor graph – with the variables representing the possible edges of the analyses.

Since well-formedness is enforced by factors, additional constraints on the shape of analyses can be imposed without changing the core algorithm of the parser. We define three additional restrictions with respect to VNs: 1) A VN that is attached to *unused* may not have any dependents. 2) A VN may not be attached to 0 if it has no dependents. 3) Only VNs may be attached to the *unused* node.

For a given sentence prefix, let A be the set of possible edges, V the set of all vertices, $N \subset V$ the VNs and $u \in V$ the *unused* node. Moreover, let $B \subset A$ be the set of edges building a well-formed analysis and $z_a \triangleq \mathbb{I}(a \in B)$, where $\mathbb{I}(\cdot)$ is the indicator function. The three additional conditions can be expressed as linear constraints which ensure that every output is a valid PDA:

$$z_{\langle n,j \rangle} + z_{\langle u,n \rangle} \leq 1, \quad n \in N, j \in V \quad (1)$$

$$z_{\langle 0,n \rangle} \leq \sum_{j \in V} z_{\langle n,j \rangle}, \quad n \in N \quad (2)$$

$$z_{\langle u,i \rangle} = 0, \quad i \in V \setminus N \quad (3)$$

The current implementation is pseudo-incremental. It reinitializes the ILP for every increment without passing intermediate results from one incremental processing step to the next, although this might be an option for further optimization.

High quality incremental parsing results can not be expected from models which have only been trained on whole-sentence annotations. If a parser is trained on gold-standard PDAs (generated as described in section 3), it would include every VN into every analysis because that data does not include any non-attached VNs. We therefore add non-attached VNs to the generated PDAs until they contain at least the set of VNs that is later used during parsing. For instance, each German training increment contains at least one virtual verb and

two virtual nouns and each English one at least one virtual verb and one virtual noun. This way, the percentage of VNs of a specific type being attached in the training data resembles the a priori probability that a VN of that type should be included by the parser while parsing.

TurboParser is trained on these extended PDAs and no adaptation of the training algorithm is needed. The training data is heavily skewed because words at the beginning of the sentences are more prevalent than the ones at the end. As a comparison with a version trained on non-incremental data shows, this has no noticeable effect on the parsing quality.

5 Evaluation

The usual methods to determine the quality of a dependency parser – labeled and unlabeled attachment scores (AS) – are not sufficient for the evaluation of incremental parsers. If the AS is computed for whole sentences, all incremental output is discarded and not considered at all. If every intermediate PDA is used, words at the start of a sentence are counted more often than the ones at the end. No information becomes available on how the accuracy of attachments evolves while parsing proceeds, and the prediction quality (i.e. the VNs) is completely ignored. Therefore, we adopt the enhanced mode of evaluation proposed by Beuck et al. (2013): In addition to the accuracy for whole sentences, the accuracies of the n newest words of each analysis are computed. This yields a curve that shows how good a word can be assumed to be attached depending on its distance to the most recent word.

Let $\langle V, G \rangle$ be the gold standard analysis of an increment and $\langle V', P \rangle$ the corresponding parser output. V and V' are the vertices and G and P the respective edges of the analyses. Let V'_p and V'_v be the in-prefix and virtual subset of V' , respectively. To evaluate the prediction capabilities of a parser, for each increment an optimal partial, surjective mapping¹ $V' \rightarrow V$ from the output produced by the parser to the (automatically generated) gold standard is computed, where each non-virtual element of V' has to be mapped to the corresponding element in V . Let M be the set of all such mappings. Then the best mapping is defined as follows:

$$\phi = \arg \max_{m \in M} \sum_{w \in V'} \mathbb{I}(\pi(m(w)) = m(\pi(w)))$$

¹The mapping is partial because for some VNs in V' there might be no corresponding VN in the gold standard.

We define a word w as correctly attached (ignoring the label) if $\pi(\phi(w)) = \phi(\pi(w))$. In an incremental analysis, an attachment of a word w can be classified into four cases:

correct $\pi(\phi(w)) = \phi(\pi(w)), \pi(w) \in V'_p$

corr. pred. $\pi(\phi(w)) = \phi(\pi(w)), \pi(w) \in V'_v$

wrong pred. $\pi(\phi(w)) \neq \phi(\pi(w)), \pi(w) \in V'_v$

wrong $\pi(\phi(w)) \neq \phi(\pi(w)), \pi(w) \in V'_p$

We can count the number of VNs that have been correctly attached: Let T be the set of all analyses produced by the parser and ϕ_t the best mapping as defined above for each $t \in T$. Furthermore, let $vn(t)$ be the set of VNs in t . The total number of correct predictions of VNs is then defined as:

$$corr = \sum_{t \in T} \sum_{v \in vn(t)} \mathbb{I}(\pi(\phi_t(v)) = \phi_t(\pi(v)))$$

Precision and recall for the prediction with VNs can be computed by dividing $corr$ by the number of predicted VNs and the number of VNs in the gold standard, respectively.

Evaluation has been carried out on the PTB converted to dependency structure using the LTH converter (Johansson and Nugues, 2007) and on the Hamburg Dependency Treebank (Foth et al., 2014). From both corpora predictive PDAs padded with unused virtual nodes have been created for training. For English, the sentences of part 1-9 of the PTB were used, for German the first 50,000 sentences of the HDT have been selected. Testing was done using one virtual noun and one virtual verb for English and two virtual nouns and one virtual verb for German because these sets cover about 90% of the prefixes in both training sets.

Figure 1 shows the evaluation results for parsing German and English using TurboParser. For both languages the attachment accuracy rises with the amount of context available. The difference between the attachment accuracy of the most recent word (relative time point 0, no word to the right of it) and the second newest word (time point 1) is strongest, especially for English. The word five elements left of the newest word (time point 5) gets attached with an accuracy that is nearly as high as the accuracy for the whole sentence (final).

The types of errors made for German and English are similar. For both German and English the unlabeled precision reaches more than 70% (see Table 1). Even the correct dependency label of upcoming words can be predicted with a fairly high precision. TurboParser parses an increment in about 0.015 seconds, which is much faster than WCDG

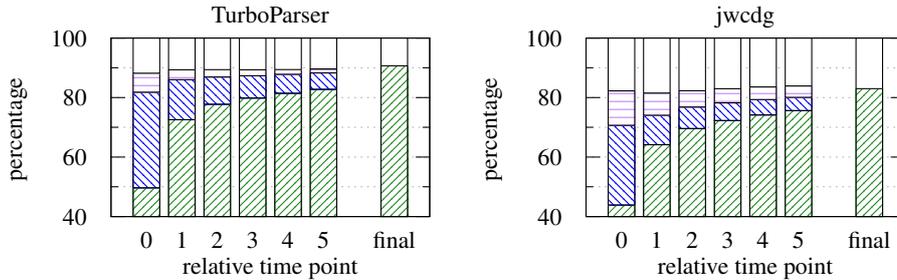


Figure 2: Results for TurboParser and jwcdg for German with tagger (labeled).

	English		German		German&tagger		German (jwcdg)	
	labeled	unlabeled	labeled	unlabeled	labeled	unlabeled	labeled	unlabeled
precision	75.47%	78.55%	67.42%	75.90%	65.21%	73.39%	32.95%	42.23%
recall	57.92%	60.29%	46.77%	52.65%	45.79%	51.54%	35.90%	46.00%

Table 1: Precision and recall for the prediction of virtual nodes

	time point 0		time point 5	
	unlabeled	labeled	unlabeled	labeled
En	89.28%	84.92%	97.32%	97.11%
De	90.91%	88.96%	96.11%	95.65%

Table 2: Stability measures

where about eight seconds per word are needed to achieve a good accuracy (Köhn and Menzel, 2013). The prediction recall is higher for English than for German which could be due to the differences in gold-standard annotation.

Training TurboParser on the non-incremental data sets results in a labeled whole-sentence accuracy of 93.02% for German. The whole-sentence accuracy for parsing with VNs is 93.33%. This shows that the additional mechanism of VNs has no negative effects on the overall parsing quality.

To compare TurboParser and WCDG running both in the predictive incremental mode, we use jwcdg, the current implementation of this approach. jwcdg differs from most other parsers in that it does not act on pre-tagged data but runs an external tagger itself in a multi-tag mode. To compare both systems, TurboParser needs to be run in a tagger-parser pipeline. We have chosen TurboTagger without look-ahead for this purpose. Running TurboParser in this pipeline leads to only slightly worse results compared to the use of gold-standard tags (see Figure 2). TurboParser’s attachment accuracy is about ten percentage points better than jwcdg’s across the board. In addition, its VN prediction is considerably better.

To measure the stability, let P_i be a prefix of the sentence P_n and a_i and a_n be the corresponding analyses produced by the parser. An attachment of a word $w \in P_i$ is stable if either w ’s head is the

same in a_i and a_n or w ’s head is not part of P_i in both a_i and a_n . The second part covers the case where the parser predicts the head of w to lie in the future and it really does, according to the final parse. Table 2 shows the attachment stability of the newest word at time point 0 compared to the word five positions to the left of time point 0. TurboParser’s stability turns out to be much higher than jwcdg’s: For German Beuck et al. (2013) report a stability of only 80% at the most recent word. Interestingly, labeling the newest attachment for English seems to be much harder than for German.

6 Conclusion

Using a parser based on ILP, we were able to analyze sentences incrementally and produce connected dependency analyses at every point in time. The intermediate structures produced by the parser are highly informative, including predictions for properties and structural embeddings of upcoming words. In contrast to previous approaches, we achieve state-of-the-art accuracy for whole sentences by abandoning strong monotonicity and aim at high stability instead, allowing the parser to improve intermediate results in light of new evidence.

The parser is trained on treebank data for whole sentences from which prefix annotations are derived in a fully automatic manner. To guide this process, a specification of structurally and lexically determined dependency relations and some additional heuristics are needed. For parsing, only a set of possible VNs has to be provided. These are the only language specific components required. Therefore, the approach can be ported to other languages with quite modest effort.

References

- Niels Beuck and Wolfgang Menzel. 2013. Structural prediction in incremental dependency parsing. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 7816 of *Lecture Notes in Computer Science*, pages 245–257. Springer Berlin Heidelberg.
- Niels Beuck, Arne Köhn, and Wolfgang Menzel. 2011. Incremental parsing and the evaluation of partial dependency analyses. In *Proceedings of the 1st International Conference on Dependency Linguistics*. Depling 2011.
- Niels Beuck, Arne Köhn, and Wolfgang Menzel. 2013. Predictive incremental parsing and its evaluation. In Kim Gerdes, Eva Hajičová, and Leo Wanner, editors, *Computational Dependency Theory*, volume 258 of *Frontiers in Artificial Intelligence and Applications*, pages 186 – 206. IOS press.
- Vera Demberg-Winterfors. 2010. *A Broad-Coverage Model of Prediction in Human Sentence Processing*. Ph.D. thesis, University of Edinburgh.
- Kilian A. Foth, Niels Beuck, Arne Köhn, and Wolfgang Menzel. 2014. The Hamburg Dependency Treebank. In *Proceedings of the Language Resources and Evaluation Conference 2014*. LREC, European Language Resources Association (ELRA).
- Hany Hassan, Khalil Sima'an, and Andy Way. 2009. Lexicalized semi-incremental dependency parsing. In *Proceedings of the International Conference RANLP-2009*, pages 128–134, Borovets, Bulgaria, September. Association for Computational Linguistics.
- Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In *Proceedings of NODALIDA 2007*, pages 105–112, Tartu, Estonia, May 25–26.
- Arne Köhn and Wolfgang Menzel. 2013. Incremental and predictive dependency parsing under real-time conditions. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 373–381, Hissar, Bulgaria, September. INCOMA Ltd. Shoumen, BULGARIA.
- Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The Penn Treebank: Annotating predicate argument structure. In *Proceedings of the Workshop on Human Language Technology, HLT '94*, pages 114–119, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Andre 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, August.
- Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.
- Davin Schlangen and Gabriel Skantze. 2011. A general, abstract model of incremental dialogue processing. *Dialogue and Discourse*, 2(1):83–111.
- Patrick Sturt and Vincenzo Lombardo. 2005. Processing coordinated structures: Incrementality and connectedness. *Cognitive Science*, 29(2):291–305.

Tailoring Continuous Word Representations for Dependency Parsing

Mohit Bansal Kevin Gimpel Karen Livescu
Toyota Technological Institute at Chicago, IL 60637, USA
{mbansal, kgimpel, klivescu}@ttic.edu

Abstract

Word representations have proven useful for many NLP tasks, e.g., Brown clusters as features in dependency parsing (Koo et al., 2008). In this paper, we investigate the use of *continuous* word representations as features for dependency parsing. We compare several popular embeddings to Brown clusters, via multiple types of features, in both news and web domains. We find that all embeddings yield significant parsing gains, including some recent ones that can be trained in a fraction of the time of others. Explicitly tailoring the representations for the task leads to further improvements. Moreover, an ensemble of all representations achieves the best results, suggesting their complementarity.

1 Introduction

Word representations derived from unlabeled text have proven useful for many NLP tasks, e.g., part-of-speech (POS) tagging (Huang et al., 2014), named entity recognition (Miller et al., 2004), chunking (Turian et al., 2010), and syntactic parsing (Koo et al., 2008; Finkel et al., 2008; Täckström et al., 2012). Most word representations fall into one of two categories. **Discrete** representations consist of memberships in a (possibly hierarchical) hard clustering of words, e.g., via k -means or the Brown et al. (1992) algorithm. **Continuous** representations (or distributed representations or embeddings) consist of low-dimensional, real-valued vectors for each word, typically induced via neural language models (Bengio et al., 2003; Mnih and Hinton, 2007) or spectral methods (Deerwester et al., 1990; Dhillon et al., 2011).

Koo et al. (2008) found improvement on in-domain dependency parsing using features based on discrete Brown clusters. In this paper, we experiment with parsing features derived from con-

tinuous representations. We find that simple attempts based on discretization of individual word vector dimensions do not improve parsing. We see gains only after first performing a hierarchical clustering of the continuous word vectors and then using features based on the hierarchy.

We compare several types of continuous representations, including those made available by other researchers (Turian et al., 2010; Collobert et al., 2011; Huang et al., 2012), and embeddings we have trained using the approach of Mikolov et al. (2013a), which is orders of magnitude faster than the others. The representations exhibit different characteristics, which we demonstrate using both intrinsic metrics and extrinsic parsing evaluation. We report significant improvements over our baseline on both the Penn Treebank (PTB; Marcus et al., 1993) and the English Web treebank (Petrov and McDonald, 2012).

While all embeddings yield some parsing improvements, we find larger gains by tailoring them to capture similarity in terms of context within syntactic parses. To this end, we use two simple modifications to the models of Mikolov et al. (2013a): a smaller context window, and conditioning on syntactic context (dependency links and labels). Interestingly, the Brown clusters of Koo et al. (2008) prove to be difficult to beat, but we find that our syntactic tailoring can lead to embeddings that match the parsing performance of Brown (on all test sets) in a fraction of the training time. Finally, a simple parser ensemble on all the representations achieves the best results, suggesting their complementarity for dependency parsing.

2 Continuous Word Representations

There are many ways to train continuous representations; in this paper, we are primarily interested in neural language models (Bengio et al., 2003), which use neural networks and local context to learn word vectors. Several researchers have made their trained representations publicly avail-

Representation	Source	Corpus	Types, Tokens	V	D	Time
BROWN	Koo et al. (2008)	BLLIP	317K, 43M	316,710	–	2.5 days [†]
SENNA	Collobert et al. (2011)	Wikipedia	8.3M, 1.8B	130,000	50	2 months*
TURIAN	Turian et al. (2010)	RCV1	269K, 37M	268,810	50	few weeks*
HUANG	Huang et al. (2012)	Wikipedia	8.3M, 1.8B	100,232	50	—
CBOW, SKIP, SKIP _{DEP}	Mikolov et al. (2013a)	BLLIP	317K, 43M	316,697	100	2-4 mins. [†]

Table 1: Details of word representations used, including datasets, vocabulary size V , and dimensionality D . Continuous representations require an additional 4 hours to run hierarchical clustering to generate features (§3.2). RCV1 = Reuters Corpus, Volume 1. * = time reported by authors. † = run by us on a 3.50 GHz desktop, using a single thread.

able, which we use directly in our experiments. In particular, we use the SENNA embeddings of Collobert et al. (2011); the scaled TURIAN embeddings (C&W) of Turian et al. (2010); and the HUANG global-context, single-prototype embeddings of Huang et al. (2012). We also use the BROWN clusters trained by Koo et al. (2008). Details are given in Table 1.

Below, we describe embeddings that we train ourselves (§2.1), aiming to make them more useful for parsing via smaller context windows (§2.1.1) and conditioning on syntactic context (§2.1.2). We then compare the representations using two intrinsic metrics (§2.2).

2.1 Syntactically-tailored Representations

We train word embeddings using the continuous bag-of-words (CBOW) and skip-gram (SKIP) models described in Mikolov et al. (2013a; 2013b) as implemented in the open-source toolkit `word2vec`. These models avoid hidden layers in the neural network and hence can be trained in only minutes, compared to days or even weeks for the others, as shown in Table 1.¹ We adapt these embeddings to be more useful for dependency parsing in two ways, described next.

2.1.1 Smaller Context Windows

The CBOW model learns vectors to predict a word given its set of surrounding context words in a window of size w . The SKIP model learns embeddings to predict each individual surrounding word given one particular word, using an analogous window size w . We find that w affects the embeddings substantially: with large w , words group with others that are topically-related; with small w , grouped words tend to share the same POS tag. We discuss this further in the intrinsic evaluation presented in §2.2.

¹We train both models on BLLIP (LDC2000T43) with PTB removed, the same corpus used by Koo et al. (2008) to train their BROWN clusters. We created a special vector for unknown words by averaging the vectors for the 50K least frequent words; we did not use this vector for the SKIP_{DEP} (§2.1.2) setting because it performs slightly better without it.

2.1.2 Syntactic Context

We expect embeddings to help dependency parsing the most when words that have similar parents and children are close in the embedding space. To target this type of similarity, we train the SKIP model on *dependency context* instead of the linear context in raw text. When ordinarily training SKIP embeddings, words v' are drawn from the neighborhood of a target word v , and the sum of log-probabilities of each v' given v is maximized. We propose to instead choose v' from the set containing the grandparent, parent, and children words of v in an automatic dependency parse.

A simple way to implement this idea is to train the original SKIP model on a corpus of dependency links and labels. For this, we parse the BLLIP corpus (minus PTB) using our baseline dependency parser, then build a corpus in which each line contains a single child word c , its parent word p , its grandparent g , and the dependency label ℓ of the $\langle c, p \rangle$ link:

$$“\ell_{\langle L \rangle} \ g_{\langle G \rangle} \ p \ c \ \ell_{\langle L \rangle}”,$$

that is, both the dependency label and grandparent word are subscripted with a special token to avoid collision with words.² We train the SKIP model on this corpus of tuples with window size $w = 1$, denoting the result SKIP_{DEP}. Note that this approach needs a parsed corpus, but there also already exist such resources (Napoles et al., 2012; Goldberg and Orwant, 2013).

2.2 Intrinsic Evaluation of Representations

Short of running end-to-end parsing experiments, how can we choose which representations to use for parsing tasks? Several methods have been proposed for intrinsic evaluation of word representa-

²We use a subscript on g so that it will be treated differently from c when considering the context of p . We removed all $g_{\langle G \rangle}$ from the vocabulary after training. We also tried adding information about POS tags. This increases M-1 (§2.2), but harms parsing performance, likely because the embeddings become too tag-like. Similar ideas have been used for clustering (Sagae and Gordon, 2009; Haffari et al., 2011; Grave et al., 2013), semantic space models (Padó and Lapata, 2007), and topic modeling (Boyd-Graber and Blei, 2008).

Representation	SIM	M-1
BROWN	–	89.3
SENNA	49.8	85.2
TURIAN	29.5	87.2
HUANG	62.6	78.1
CROWD		
CROWD, $w = 2$	34.7	84.8
SKIP, $w = 1$	37.8	86.6
SKIP, $w = 2$	43.1	85.8
SKIP, $w = 5$	44.4	81.1
SKIP, $w = 10$	44.6	71.5
SKIP _{DEP}	34.6	88.3

Table 2: Intrinsic evaluation of representations. SIM column has Spearman’s $\rho \times 100$ for 353-pair word similarity dataset. M-1 is our unsupervised POS tagging metric. For BROWN, M-1 is simply many-to-one accuracy of the clusters. Best score in each column is bold.

tions; we discuss two here:

Word similarity (SIM): One widely-used evaluation compares distances in the continuous space to human judgments of word similarity using the 353-pair dataset of Finkelstein et al. (2002). We compute cosine similarity between the two vectors in each word pair, then order the word pairs by similarity and compute Spearman’s rank correlation coefficient (ρ) with the gold similarities. Embeddings with high ρ capture similarity in terms of paraphrase and topical relationships.

Clustering-based tagging accuracy (M-1): Intuitively, we expect embeddings to help parsing the most if they can tell us when two words are similar *syntactically*. To this end, we use a metric based on unsupervised evaluation of POS taggers. We perform clustering and map each cluster to one POS tag so as to maximize tagging accuracy, where multiple clusters can map to the same tag. We cluster vectors corresponding to the tokens in PTB WSJ sections 00-21.³

Table 2 shows these metrics for representations used in this paper. The BROWN clusters have the highest M-1, indicating high cluster purity in terms of POS tags. The HUANG embeddings have the highest SIM score but low M-1, presumably because they were trained with global context, making them more tuned to capture topical similarity. We compare several values for the window size (w) used when training the SKIP embeddings, finding that small w leads to higher M-1 and lower SIM. Table 3 shows examples of clusters obtained by clustering SKIP embeddings of $w = 1$ versus $w = 10$, and we see that the former correspond closely to POS tags, while the latter are

³For clustering, we use k -means with $k = 1000$ and initialize by placing centroids on the 1000 most-frequent words.

w	Example clusters
1	[Mr., Mrs., Ms., Prof., ...], [Jeffrey, Dan, Robert, Peter, ...], [Johnson, Collins, Schmidt, Freedman, ...], [Portugal, Iran, Cuba, Ecuador, ...], [CST, 4:30, 9-10:30, CDT, ...], [his, your, her, its, ...], [truly, wildly, politically, financially, ...]
10	[takeoff, altitude, airport, carry-on, airplane, flown, landings, ...], [health-insurance, clinic, physician, doctor, medical, health-care, ...], [financing, equity, investors, firms, stock, fund, market, ...]

Table 3: Example clusters for SKIP embeddings with window size $w = 1$ (syntactic) and $w = 10$ (topical).

much more topically-coherent and contain mixed POS tags.⁴ For parsing experiments, we choose $w = 2$ for CROWD and $w = 1$ for SKIP. Finally, our SKIP_{DEP} embeddings, trained with syntactic context and $w = 1$ (§2.1.2), achieve the highest M-1 of all continuous representations. In §4, we will relate these intrinsic metrics to extrinsic parsing performance.

3 Dependency Parsing Features

We now discuss the features that we add to our baseline dependency parser (second-order MST-Parser; McDonald and Pereira, 2006) based on discrete and continuous representations.

3.1 Brown Cluster Features

We start by replicating the features of Koo et al. (2008) using their BROWN clusters; each word is represented by a 0-1 bit string indicating the path from the root to the leaf in the binary merge tree. We follow Koo et al. in adding cluster versions of the first- and second-order features in MSTParser, using bit string prefixes of the head, argument, sibling, intermediate words, etc., to augment or replace the POS and lexical identity information. We tried various sets of prefix lengths on the development set and found the best setting to use prefixes of length 4, 6, 8, and 12.⁵

3.2 Continuous Representation Features

We tried two kinds of indicator features:

Bucket features: For both parent and child vectors in a potential dependency, we fire one indicator feature per dimension of each embedding

⁴A similar effect, when changing distributional context window sizes, was found by Lin and Wu (2009).

⁵See Koo et al. (2008) for the exact feature templates. They used the full string in place of the length-12 prefixes, but that setting worked slightly worse for us. Note that the baseline parser used by Koo et al. (2008) is different from the second-order MSTParser that we use here; their parser allows grandparent interactions in addition to the sibling interactions in ours. We use their clusters, available at <http://people.csail.mit.edu/maestro/papers/blip-clusters.gz>.

vector, where the feature consists of the dimension index d and a bucketed version of the embedding value in that dimension, i.e., $bucket_k(E_{vd})$ for word index v and dimension d , where E is the $V \times D$ embedding matrix.⁶ We also tried standard conjunction variants of this feature consisting of the bucket values of both the head and argument along with their POS-tag or word information, and the attachment distance and direction.⁷

Cluster bit string features: To take into account all dimensions simultaneously, we perform agglomerative hierarchical clustering of the embedding vectors. We use Ward’s minimum variance algorithm (Ward, 1963) for cluster distance and the Euclidean metric for vector distance (via MATLAB’s `linkage` function with `{method=ward, metric=euclidean}`). Next, we fire features on the hierarchical clustering bit strings using templates identical to those for BROWN, except that we use longer prefixes as our clustering hierarchies tend to be deeper.⁸

4 Parsing Experiments

Setup: We use the publicly-available MST-Parser for all experiments, specifically its second-order projective model.⁹ We remove all features that occur only once in the training data. For WSJ parsing, we use the standard train(02-21)/dev(22)/test(23) split and apply the NP bracketing patch by Vadas and Curran (2007). For Web parsing, we still train on WSJ 02-21, but test on the five Web domains (answers, email, newsgroup, reviews, and weblog) of the ‘English Web Treebank’ (LDC2012T13), splitting each domain in half (in original order) for the development and test sets.¹⁰ For both treebanks, we convert from constituent to dependency format using `pennconverter` (Johansson and Nugues, 2007), and generate POS tags using the MXPOST tagger (Ratnaparkhi, 1996). To evaluate, we use

⁶Our bucketing function $bucket_k(x)$ converts the real value x to its closest multiple of k . We choose a k value of around 1/5th of the embedding’s absolute range.

⁷We initially experimented directly with real-valued features (instead of bucketed indicator features) and similar conjunction variants, but these did not perform well.

⁸We use prefixes of length 4, 6, 8, 12, 16, 20, and full-length, again tuned on the development set.

⁹We use the recommended MSTParser settings: training-k:5 iters:10 loss-type:nopunc decode-type:proj

¹⁰Our setup is different from SANCL 2012 (Petrov and McDonald, 2012) because the exact splits and test data were only available to participants.

System	Dev	Test
Baseline	92.38	91.95
BROWN	93.18	92.69
SENNA (Buckets)	92.64	92.04
SENNA (Bit strings)	92.88	92.30
HUANG (Buckets)	92.44	91.86
HUANG (Bit strings)	92.55	92.36
CBOW (Buckets)	92.57	91.93
CBOW (Bit strings)	93.06	92.53

Table 4: Bucket vs. bit string features (UAS on WSJ).

System	Dev	Test
Baseline	92.38	91.95
BROWN	93.18	92.69
SENNA	92.88	92.30
TURIAN	92.84	92.26
HUANG	92.55	92.36
CBOW	93.06	92.53
SKIP	92.94	92.29
SKIP _{DEP}	93.33	92.69
Ensemble Results		
ALL – BROWN	93.46	92.90
ALL	93.54	92.98

Table 5: Full results with bit string features (UAS on WSJ).

unlabeled attachment score (UAS).¹¹ We report statistical significance ($p < 0.01$, 100K samples) using the bootstrap test (Efron and Tibshirani, 1994).

Comparing bucket and bit string features: In Table 4, we find that bucket features based on individual embedding dimensions do not lead to improvements in test accuracy, while bit string features generally do. This is likely because individual embedding dimensions rarely correspond to interpretable or useful distinctions among words, whereas the hierarchical bit strings take into account all dimensions of the representations simultaneously. Their prefixes also naturally define features at multiple levels of granularity.

WSJ results: Table 5 shows our main WSJ results. Although BROWN yields one of the highest individual gains, we also achieve statistically significant gains over the baseline from all embeddings. The CBOW embeddings perform as well as BROWN (i.e., no statistically significant difference) but are orders of magnitude faster to train. Finally, the syntactically-trained SKIP_{DEP} embeddings are statistically indistinguishable from BROWN and CBOW, and significantly better than all other embeddings. This suggests that targeting the similarity captured by syntactic context is useful for dependency parsing.

¹¹We find similar improvements under labeled attachment score (LAS). We ignore punctuation : , “ ” . in our evaluation (Yamada and Matsumoto, 2003; McDonald et al., 2005).

System	ans	eml	nwg	rev	blog	Avg
Baseline	82.6	81.2	84.3	83.8	85.5	83.5
BROWN	83.4	81.7	85.2	84.5	86.1	84.2
SENNA	83.7	81.9	85.0	85.0	86.0	84.3
TURIAN	83.0	81.5	85.0	84.1	85.7	83.9
HUANG	83.1	81.8	85.1	84.7	85.9	84.1
CBOW	82.9	81.3	85.2	83.9	85.8	83.8
SKIP	83.1	81.1	84.7	84.1	85.4	83.7
SKIP _{DEP}	83.3	81.5	85.2	84.3	86.0	84.1
Ensemble Results						
ALL-BR	83.9	82.2	85.9	85.0	86.6	84.7
ALL	84.2	82.3	85.9	85.1	86.8	84.9

Table 6: Main UAS test results on Web treebanks. Here, ans=answers, eml=email, nwg=newsgroup, rev=reviews, blog=weblog, BR=BROWN, Avg=Macro-average.

Web results: Table 6 shows our main Web results.¹² Here, we see that the SENNA, BROWN, and SKIP_{DEP} embeddings perform the best on average (and are statistically indistinguishable, except SENNA vs. SKIP_{DEP} on the reviews domain). They yield statistically significant UAS improvements over the baseline across all domains, except weblog for SENNA (narrowly misses significance, $p=0.014$) and email for SKIP_{DEP}.¹³

Ensemble results: When analyzing errors, we see differences among the representations, e.g., BROWN does better at attaching proper nouns, prepositions, and conjunctions, while CBOW does better on plural common nouns and adverbs. This suggests that the representations might be complementary and could benefit from combination. To test this, we use a simple ensemble parser that chooses the highest voted parent for each argument.¹⁴ As shown in the last two rows of Tables 5 and 6, this leads to substantial gains. The ‘ALL – BROWN’ ensemble combines votes from all non-BROWN continuous representations, and the ‘ALL’ ensemble also includes BROWN.

Characteristics of representations: We now relate the intrinsic metrics from §2.2 to parsing performance. The clearest correlation appears when comparing variations of a single model, e.g., for SKIP, the WSJ dev accuracies are 93.33 (SKIP_{DEP}), 92.94 ($w = 1$), 92.86 ($w = 5$), and 92.70 ($w = 10$), which matches the M-1 score order and is the reverse of the SIM score order.

¹²We report individual domain results and macro-average over domains. We do not tune any features/parameters on Web dev sets; we only show the test results for brevity.

¹³Note that SENNA and HUANG are trained on Wikipedia which may explain why they work better on Web parsing as compared to WSJ parsing.

¹⁴This does not guarantee a valid tree. Combining *features* from representations will allow training to weigh them appropriately and also guarantee a tree.

5 Related Work

In addition to work mentioned above, relevant work that uses discrete representations exists for POS tagging (Ritter et al., 2011; Owoputi et al., 2013), named entity recognition (Ratinov and Roth, 2009), supersense tagging (Grave et al., 2013), grammar induction (Spitkovsky et al., 2011), constituency parsing (Finkel et al., 2008), and dependency parsing (Tratz and Hovy, 2011). Continuous representations in NLP have been evaluated for their ability to capture syntactic and semantic word similarity (Huang et al., 2012; Mikolov et al., 2013a; Mikolov et al., 2013b) and used for tasks like semantic role labeling, part-of-speech tagging, NER, chunking, and sentiment classification (Turian et al., 2010; Collobert et al., 2011; Dhillon et al., 2012; Al-Rfou’ et al., 2013).

For dependency parsing, Hisamoto et al. (2013) also used embedding features, but there are several differences between their work and ours. First, they use only one set of pre-trained embeddings (TURIAN) while we compare several and also train our own, tailored to the task. Second, their embedding features are simpler than ours, only using flat (non-hierarchical) cluster IDs and binary strings obtained via sign quantization ($\mathbb{1}[x > 0]$) of the vectors. They also compare to a first-order baseline and only evaluate on the Web treebanks.

Concurrently, Andreas and Klein (2014) investigate the use of embeddings in constituent parsing. There are several differences: we work on dependency parsing, use clustering-based features, and tailor our embeddings to dependency-style syntax; their work additionally studies vocabulary expansion and relating in-vocabulary words via embeddings.

6 Conclusion

We showed that parsing features based on hierarchical bit strings work better than those based on discretized individual embedding values. While the Brown clusters prove to be well-suited to parsing, we are able to match their performance with our SKIP_{DEP} embeddings that train much faster. Finally, we found the various representations to be complementary, enabling a simple ensemble to perform best. Our SKIP_{DEP} embeddings and bit strings are available at ttic.edu/bansal/data/syntacticEmbeddings.zip.

References

- Rami Al-Rfou', Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual NLP. In *Proceedings of CoNLL*.
- Jacob Andreas and Dan Klein. 2014. How much do word embeddings encode about syntax? In *Proceedings of ACL*.
- Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, March.
- Jordan L. Boyd-Graber and David M. Blei. 2008. Syntactic topic models. In *Proceedings of NIPS*.
- Peter F. Brown, Peter V. Desouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based N-gram models of natural language. *Computational Linguistics*, 18(4):467–479.
- 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.
- Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407.
- Paramveer Dhillon, Dean P. Foster, and Lyle H. Ungar. 2011. Multi-view learning of word embeddings via CCA. In *Proceedings of NIPS*.
- Paramveer Dhillon, Jordan Rodu, Dean P. Foster, and Lyle H. Ungar. 2012. Two Step CCA: A new spectral method for estimating vector models of words. In *Proceedings of ICML*.
- Bradley Efron and Robert J. Tibshirani. 1994. *An introduction to the bootstrap*, volume 57. CRC press.
- Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. 2008. Efficient, feature-based, conditional random field parsing. In *Proceedings of ACL*.
- Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.
- Yoav Goldberg and Jon Orwant. 2013. A dataset of syntactic-ngrams over time from a very large corpus of English books. In *Second Joint Conference on Lexical and Computational Semantics (*SEM)*, volume 1, pages 241–247.
- Edouard Grave, Guillaume Obozinski, and Francis Bach. 2013. Hidden markov tree models for semantic class induction. In *Proceedings of CoNLL*.
- Gholamreza Haffari, Marzieh Razavi, and Anoop Sarkar. 2011. An ensemble model that combines syntactic and semantic clustering for discriminative dependency parsing. In *Proceedings of ACL*.
- Sorami Hisamoto, Kevin Duh, and Yuji Matsumoto. 2013. An empirical investigation of word representations for parsing the web. In *Proceedings of ANLP*.
- Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving Word Representations via Global Context and Multiple Word Prototypes. In *Proceedings of ACL*.
- Fei Huang, Arun Ahuja, Doug Downey, Yi Yang, Yuhong Guo, and Alexander Yates. 2014. Learning representations for weakly supervised natural language processing tasks. *Computational Linguistics*, 40(1).
- Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In *16th Nordic Conference of Computational Linguistics*.
- Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL*.
- Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proceedings of ACL-IJCNLP*.
- Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.
- Ryan T. McDonald and Fernando C. Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL*.
- Ryan T. McDonald, Koby Crammer, and Fernando C. Pereira. 2005. Spanning tree methods for discriminative training of dependency parsers. Technical Report MS-CIS-05-11, University of Pennsylvania.
- 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 Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*.
- Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *Proceedings of HLT-NAACL*.
- Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of ICML*.

- 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 '12*, pages 95–100, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Olutobi Owoputi, Brendan OConnor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of NAACL*.
- Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.
- Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web. In *Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL)*.
- Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of CoNLL*.
- Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of EMNLP*.
- Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of EMNLP*.
- Kenji Sagae and Andrew S. Gordon. 2009. Clustering words by syntactic similarity improves dependency parsing of predicate-argument structures. In *Proceedings of the 11th International Conference on Parsing Technologies*.
- Valentin I. Spitzkovsky, Hiyan Alshawi, Angel X. Chang, and Daniel Jurafsky. 2011. Unsupervised dependency parsing without gold part-of-speech tags. In *Proceedings of EMNLP*.
- Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of NAACL*.
- Stephen Tratz and Eduard Hovy. 2011. A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of EMNLP*.
- Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of ACL*.
- David Vadas and James R. Curran. 2007. Adding noun phrase structure to the Penn Treebank. In *Proceedings of ACL*.
- Joe H. Ward. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.
- Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of International Conference on Parsing Technologies*.

Observational Initialization of Type-Supervised Taggers

Hui Zhang*

Department of Computer Science
University of Southern California
hzhang@isi.edu

John DeNero

Google, Inc.
denero@google.com

Abstract

Recent work has sparked new interest in type-supervised part-of-speech tagging, a data setting in which no labeled sentences are available, but the set of allowed tags is known for each word type. This paper describes *observational initialization*, a novel technique for initializing EM when training a type-supervised HMM tagger. Our initializer allocates probability mass to unambiguous transitions in an unlabeled corpus, generating token-level observations from type-level supervision. Experimentally, observational initialization gives state-of-the-art type-supervised tagging accuracy, providing an error reduction of 56% over uniform initialization on the Penn English Treebank.

1 Introduction

For many languages, there exist comprehensive dictionaries that list the possible parts-of-speech for each word type, but there are no corpora labeled with the part-of-speech of each token in context. Type-supervised tagging (Merialdo, 1994) explores this scenario; a model is provided with type-level information, such as the fact that “only” can be an adjective, adverb, or conjunction, but not any token-level information about which instances of “only” in a corpus are adjectives. Recent research has focused on using type-level supervision to infer token-level tags. For instance, Li et al. (2012) derive type-level supervision from Wiktionary, Das and Petrov (2011) and Täckström et al. (2013) project type-level tag sets across languages, and Garrette and Baldrige (2013) solicit type-level annotations directly from speakers. In all of these efforts, a probabilistic sequence model is trained to disambiguate token-level tags that are

constrained to match type-level tag restrictions. This paper describes *observational initialization*, a simple but effective learning technique for training type-supervised taggers.

A hidden Markov model (HMM) can be used to disambiguate tags of individual tokens by maximizing corpus likelihood using the expectation maximization (EM) algorithm. Our approach is motivated by a suite of oracle experiments that demonstrate the effect of initialization on the final tagging accuracy of an EM-trained HMM tagger. We show that initializing EM with accurate transition model parameters is sufficient to guide learning toward a high-accuracy final model.

Inspired by this finding, we introduce *observational initialization*, which is a simple method to heuristically estimate transition parameters for a corpus using type-level supervision. Transition probabilities are estimated from unambiguous consecutive tag pairs that arise when two consecutive words each have only a single allowed tag. These unambiguous word pairs can be tagged correctly without any statistical inference. Initializing EM with the relative frequency of these unambiguous pairs improves tagging accuracy dramatically over uniform initialization, reducing errors by 56% in English and 29% in German. This efficient and data-driven approach gives the best reported tagging accuracy for type-supervised sequence models, outperforming the minimized model of Ravi and Knight (2009), the Bayesian LDA-based model of Toutanova and Johnson (2008), and an HMM trained with language-specific initialization described by Goldberg et al. (2008).

2 Type-Supervised Tagging

A first-order Markov model for part-of-speech tagging defines a distribution over sentences for which a single tag is given to each word token. Let $w_i \in W$ refer to the i th word in a sentence w , drawn from language vocabulary W . Likewise,

*Research conducted during an internship at Google.

$t_i \in T$ is the tag in tag sequence \mathbf{t} of the i th word, drawn from tag inventory T . The joint probability of a sentence can be expressed in terms of two sets of parameters for conditional multinomial distributions: ϕ defines the probability of a tag given its previous tag and θ defines the probability of a word given its tag.

$$P_{\phi,\theta}(\mathbf{w}, \mathbf{t}) = \prod_{i=1}^{|\mathbf{w}|} P_{\phi}(t_i|t_{i-1}) \cdot P_{\theta}(w_i|t_i)$$

Above, t_0 is a fixed start-of-sentence tag.

For a set of sentences \mathcal{S} , the EM algorithm can be used to iteratively find a local maximum of the corpus log-likelihood:

$$\ell(\phi, \theta; \mathcal{S}) = \sum_{\mathbf{w} \in \mathcal{S}} \ln \left[\sum_{\mathbf{t}} P_{\phi,\theta}(\mathbf{w}, \mathbf{t}) \right]$$

The parameters ϕ and θ can then be used to predict the most likely sequence of tags for each sentence under the model:

$$\hat{\mathbf{t}}(\mathbf{w}) = \arg \max_{\mathbf{t}} P_{\phi,\theta}(\mathbf{w}, \mathbf{t})$$

Tagging accuracy is the fraction of these tags in $\hat{\mathbf{t}}(\mathbf{w})$ that match hand-labeled oracle tags $\mathbf{t}^*(\mathbf{w})$.

Type Supervision. In addition to an unlabeled corpus of sentences, type-supervised models also have access to a tag dictionary $D \subseteq W \times T$ that contains all allowed word-tag pairs. For an EM-trained HMM, initially setting $P_{\theta}(w|t) = 0$ for all $(w, t) \notin D$ ensures that all words will be labeled with allowed tags.

Tag dictionaries can be derived from various sources, such as lexicographic resources (Li et al., 2012) and cross-lingual projections (Das and Petrov, 2011). In this paper, we will follow previous work in deriving the tag dictionary from a labeled corpus (Smith and Eisner, 2005); this synthetic setting maximizes experiment repeatability and allows for direct comparison of type-supervised learning techniques.

Transductive Applications. We consider a transductive data setting in which the test set is available during training. In this case, the model is not required to generalize to unseen examples or unknown words, as in the typical inductive setting.

Transductive learning arises in document clustering and corpus analysis applications. For example, before running a document clustering algorithm on a fixed corpus of documents, it may be

useful to tag each word with its most likely part-of-speech in context, disambiguating the lexical features in a bag-of-words representation. In corpus analysis or genre detection, it may be useful to determine for a fixed corpus the most common part-of-speech for each word type, which could be inferred by tagging each word with its most likely part-of-speech. In both cases, the set of sentences to tag is known in advance of learning.

3 Initializing HMM Taggers

The EM algorithm is sensitive to initialization. In a latent variable model, different parameter values may yield similar data likelihoods but very different predictions. We explore this issue via experiments on the Wall Street Journal section of the English Penn Treebank (Marcus et al., 1993). We adopt the transductive data setting introduced by Smith and Eisner (2005) and used by Goldwater and Griffiths (2007), Toutanova and Johnson (2008) and Ravi and Knight (2009); models are trained on all sections 00-24, the tag dictionary D is constructed by allowing all word-tag pairs appearing in the entire labeled corpus, and the tagging accuracy is evaluated on a 1005 sentence subset sampled from the corpus.

The degree of variation in tagging accuracy due to initialization can be observed most clearly by two contrasting initializations. UNIFORM initializes the model with uniform distributions over allowed outcomes:

$$P_{\phi}(t|t') = \frac{1}{|T|}$$

$$P_{\theta}(w|t) = \frac{1}{|\{w : (w, t) \in D\}|}$$

SUPERVISED is an oracle setting that initializes the model with the relative frequency of observed pairs in a labeled corpus:

$$P_{\phi}(t|t') \propto \sum_{(\mathbf{w}, \mathbf{t}^*)} \sum_{i=1}^{|\mathbf{w}|} \delta((t_i^*, t_{i-1}^*), (t, t'))$$

$$P_{\theta}(w|t) \propto \sum_{(\mathbf{w}, \mathbf{t}^*)} \sum_{i=1}^{|\mathbf{w}|} \delta((w_i, t_i^*), (w, t))$$

where the Kronecker $\delta(x, y)$ function is 1 if x and y are equal and 0 otherwise.

Figure 1 shows that while UNIFORM and SUPERVISED achieve nearly identical data log-likelihoods, their final tagging accuracy differs by

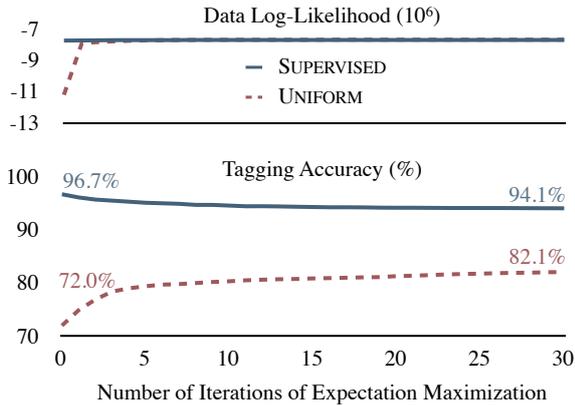


Figure 1: The data log-likelihood (top) and tagging accuracy (bottom) of two contrasting initializers, UNIFORM and SUPERVISED, compared on the Penn Treebank.

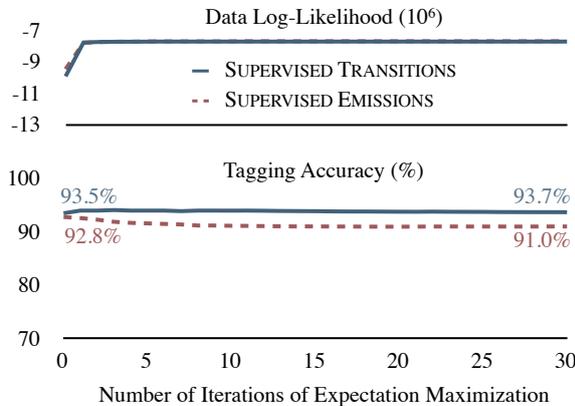


Figure 2: The data log-likelihood (top) and tagging accuracy (bottom) of two partially supervised initializers, one with SUPERVISED TRANSITIONS and one with SUPERVISED EMISSIONS, compared on the Penn Treebank.

12%. Accuracy degrades somewhat from the SUPERVISED initialization, since the data likelihood objective differs from the objective of maximizing tagging accuracy. However, the final SUPERVISED performance of 94.1% shows that there is substantial room for improvement over the UNIFORM initializer.

Figure 2 compares two partially supervised initializations. SUPERVISED TRANSITIONS initializes the transition model with oracle counts, but the emission model uniformly. Conversely, SUPERVISED EMISSIONS initializes the emission parameters from oracle counts, but initializes the transition model uniformly. There are many more emission parameters (57,390) than transition parameters (1,858). Thus, it is not surprising that

SUPERVISED EMISSIONS gives a higher initial likelihood. Again, both initializers lead to solutions with nearly the same likelihood as SUPERVISED and UNIFORM.

Figure 2 shows that SUPERVISED TRANSITIONS outperforms SUPERVISED EMISSIONS in tagging accuracy, despite the fact that fewer parameters are set with supervision. With fixed D , an accurate initialization of the transition distributions leads to accurate tagging after EM training. We therefore concentrate on developing an effective initialization for the transition distribution.

4 Observational Initialization

The SUPERVISED TRANSITIONS initialization is estimated from observations of consecutive tags in a labeled corpus. Our OBSERVATIONAL initializer is likewise estimated from the relative frequency of consecutive tags, taking advantage of the structure of the tag dictionary D . However, it does not require a labeled corpus.

Let $D(w, \cdot) = \{t : (w, t) \in D\}$ denote the allowed tags for word w . The set

$$U = \{w : |D(w, \cdot)| = 1\}$$

contains all words that have only one allowed tag. When a token of some $w \in U$ is observed in a corpus, its tag is unambiguous. Therefore, its tag is observed as well, and a portion of the tag sequence is known. When consecutive pairs of tokens are both in U , we can observe a transition in the latent tag sequence. The OBSERVATIONAL initializer simply estimates a transition distribution from the relative frequency of these unambiguous observations that occur whenever two consecutive tokens both have a unique tag.

We now formally define the observational initializer. Let $g(w, t) = \delta(D(w, \cdot), \{t\})$ be an indicator function that is 1 whenever $w \in U$ and its single allowed tag is t , and 0 otherwise. Then, we initialize ϕ such that:

$$P_\phi(t|t') \propto \sum_{w \in S} \sum_{i=1}^{|w|} g(w_i, t) \cdot g(w_{i-1}, t')$$

The emission parameters θ are set to be uniform over allowed words for each tag, as in UNIFORM initialization.

Figure 3 compares the OBSERVATIONAL initializer to the SUPERVISED TRANSITIONS initializer, and the top of Table 1 summarizes the performance of all initializers discussed so far for the

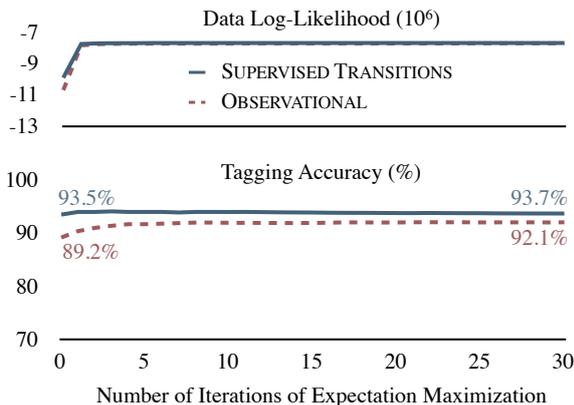


Figure 3: The data log-likelihood (top) and tagging accuracy (bottom) of initializing with SUPERVISED TRANSITIONS compared to the unsupervised OBSERVATIONAL initialization that requires only a tag dictionary and an unlabeled training corpus.

English Penn Treebank. The OBSERVATIONAL initializer provides an error reduction over UNIFORM of 56%, surpassing the performance of an initially supervised emission model and nearing the performance of a supervised transition model.

The bottom of Table 1 shows a similar comparison on the Tübingen treebank of spoken German (Telljohann et al., 2006). Both training and testing were performed on the entire treebank. The observational initializer provides an error reduction over UNIFORM of 29%, and again outperforms SUPERVISED EMISSIONS. On this dataset OBSERVATIONAL initialization matches the final performance of SUPERVISED TRANSITIONS.

5 Discussion

The fact that observations and prior knowledge are useful for part-of-speech tagging is well understood (Brill, 1995), but the approach of estimating an initial transition model only from unambiguous word pairs is novel.

Our experiments show that for EM-trained HMM taggers in a type-supervised transductive data setting, observational initialization is an effective technique for guiding training toward high-accuracy solutions, approaching the oracle accuracy of SUPERVISED TRANSITIONS initialization.

The fact that models with similar data likelihood can vary dramatically in accuracy has been observed in other learning problems. For instance, Toutanova and Galley (2011) show that optimal

English	Initial	EM-trained
UNIFORM	72.0	82.1
OBSERVATIONAL	89.2	92.1
SUP. EMISSIONS	92.8	91.0
SUP. TRANSITIONS	93.5	93.7
FULLY SUPERVISED	96.7	94.1
German	Initial	EM-trained
UNIFORM	77.2	88.8
OBSERVATIONAL	92.7	92.1
SUP. EMISSIONS	90.7	89.0
SUP. TRANSITIONS	94.8	92.0
FULLY SUPERVISED	97.0	92.9

Table 1: Accuracy of English (top) and German (bottom) tagging models at initialization (left) and after 30 iterations of EM training (right) using various initializers.

parameters for IBM Model 1 are not unique, and alignments predicted from different optimal parameters vary significantly in accuracy.

However, the effectiveness of observational initialization is somewhat surprising because EM training includes these unambiguous tag pairs in its expected counts, even with uniform initialization. Our experiments indicate that this signal is not used effectively unless explicitly encoded in the initialization.

In our English data, 48% of tokens and 74% of word types have only one allowed tag. 28% of pairs of adjacent tokens have only one allowed tag pair and contribute to observational initialization. In German, 49% of tokens and 87% of word types are unambiguous, and 26% of adjacent token pairs are unambiguous.

6 Related Work

We now compare with several previous published results on type-supervised part-of-speech tagging trained using the same data setting on the English WSJ Penn Treebank, introduced by Smith and Eisner (2005).

Contrastive estimation (Smith and Eisner, 2005) is a learning technique that approximates the partition function of the EM objective in a log-linear model by considering a neighborhood around observed training examples. The Bayesian HMM of Goldwater and Griffiths (2007) is a second-order HMM (*i.e.*, likelihood factors over triples of tags) that is estimated using a prior distribution that promotes sparsity. Sparse priors have

	45 tag set		17 tag set	
	All train	973k train	All train	973k train
Observational initialization (this work)	92.1	92.8	93.9	94.8
Contrastive Estimation (Smith and Eisner, 2005)	–	–	88.7	–
Bayesian HMM (Goldwater and Griffiths, 2007)	86.8	–	87.3	–
Bayesian LDA-HMM (Toutanova and Johnson, 2008)	–	–	93.4	–
Linguistic initialization (Goldberg et al., 2008)	91.4	–	93.8	–
Minimal models (Ravi and Knight, 2009)	–	92.3	–	96.8

Table 2: Tagging accuracy of different approaches on English Penn Treebank. Columns labeled *973k train* describe models trained on the subset of 973k tokens used by Ravi and Knight (2009).

been motivated empirically for this task (Johnson, 2007). The Bayesian HMM model predicts tag sequences via Gibbs sampling, integrating out model parameters. The Bayesian LDA-based model of Toutanova and Johnson (2008) models ambiguity classes of words, which allows information sharing among words in the tag dictionary. In addition, it incorporates morphology features and a sparse prior of tags for a word. Inference approximations are required to predict tags, integrating out model parameters.

Ravi and Knight (2009) employs integer linear programming to select a minimal set of parameters that can generate the test sentences, followed by EM to set parameter values. This technique requires the additional information of which sentences will be used for evaluation, and its scalability is limited. In addition, this work used a subset of the WSJ Penn Treebank for training and selecting a tag dictionary. This restriction actually tends to improve performance, because a smaller tag dictionary further constrains model optimization. We compare directly to their training set, kindly provided to us by the authors.

The linguistic initialization of Goldberg et al. (2008) is most similar to the current work, in that it estimates maximum likelihood parameters of an HMM using EM, but starting with a well-chosen initialization with language specific linguistic knowledge. That work estimates emission distributions using a combination of suffix morphology rules and corpus context counts.

Table 2 compares our results to these related techniques. Each column represents a variant of the experimental setting used in prior work. Smith and Eisner (2005) introduced a mapping from the full 45 tag set of the Penn Treebank to 17 coarse tags. We report results on this coarse set by projecting from the full set after learning and infer-

ence.¹ Using the full tag set or the full training data, our method offers the best published performance without language-specific assumptions or approximate inference.

7 Future Work

This paper has demonstrated a simple and effective learning method for type-supervised, transductive part-of-speech tagging. However, it is an open question whether the technique is as effective for tag dictionaries derived from more natural sources than the labels of an existing treebank.

All of the methods to which we compare except Goldberg et al. (2008) focus on learning and modeling techniques, while our method only addresses initialization. We look forward to investigating whether our technique can be used as an initialization or prior for these other methods.

References

- Eric Brill. 1995. Unsupervised learning of disambiguation rules for part of speech tagging. In *Natural Language Processing Using Very Large Corpora*, pages 1–13. Kluwer Academic Press.
- Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the Association for Computational Linguistics*.
- Dan Garrette and Jason Baldridge. 2013. Learning a part-of-speech tagger from two hours of annotation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.
- Yoav Goldberg, Meni Adler, and Michael Elhadad. 2008. EM can find pretty good HMM POS-taggers

¹Training with the reduced tag set led to lower performance of 91.0% accuracy, likely because the coarse projection drops critical information about allowable English transitions, such as what verb forms can follow *to be* (Goldberg et al., 2008).

- (when given a good start). In *Proceedings of the Association for Computational Linguistics*.
- Sharon Goldwater and Tom Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the Association for Computational Linguistics*.
- Mark Johnson. 2007. Why doesnt EM nd good HMM POS-taggers? In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Shen Li, João V. Graça, and Ben Taskar. 2012. Wiki-ly supervised part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*.
- Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational Linguistics*.
- Sujith Ravi and Kevin Knight. 2009. Minimized models for unsupervised part-of-speech tagging. In *Proceedings of the Association for Computational Linguistics*.
- Noah A. Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the Association for Computational Linguistics*.
- Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. *Transactions of the Association for Computational Linguistics*.
- Heike Telljohann, Erhard Hinrichs, Sandra Kübler, and Heike Zinsmeister. 2006. Stylebook for the tbingen treebank of written german.
- Kristina Toutanova and Michel Galley. 2011. Why initialization matters for ibm model 1: Multiple optima and non-strict convexity. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 461–466, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Kristina Toutanova and Mark Johnson. 2008. A Bayesian LDA-based model for semi-supervised part-of-speech tagging. In *Proceedings of Neural and Information Processing Systems*.

How much do word embeddings encode about syntax?

Jacob Andreas and Dan Klein
Computer Science Division
University of California, Berkeley
{jda, klein}@cs.berkeley.edu

Abstract

Do continuous word embeddings encode any useful information for constituency parsing? We isolate three ways in which word embeddings might augment a state-of-the-art statistical parser: by connecting out-of-vocabulary words to known ones, by encouraging common behavior among related in-vocabulary words, and by directly providing features for the lexicon. We test each of these hypotheses with a targeted change to a state-of-the-art baseline. Despite small gains on extremely small supervised training sets, we find that extra information from embeddings appears to make little or no difference to a parser with adequate training data. Our results support an overall hypothesis that word embeddings import syntactic information that is ultimately redundant with distinctions learned from tree-banks in other ways.

1 Introduction

This paper investigates a variety of ways in which word embeddings might augment a constituency parser with a discrete state space. Word embeddings—representations of lexical items as points in a real vector space—have a long history in natural language processing, going back at least as far as work on latent semantic analysis (LSA) for information retrieval (Deerwester et al., 1990). While word embeddings can be constructed directly from surface distributional statistics, as in LSA, more sophisticated tools for unsupervised extraction of word representations have recently gained popularity (Collobert et al., 2011; Mikolov et al., 2013a). Semi-supervised and unsupervised models for a variety of core NLP tasks, including named-entity recognition (Freitag, 2004), part-of-speech tagging (Schütze, 1995), and chunking

(Turian et al., 2010) have been shown to benefit from the inclusion of word embeddings as features. In the other direction, access to a syntactic parse has been shown to be useful for constructing word embeddings for phrases compositionally (Hermann and Blunsom, 2013; Andreas and Ghahramani, 2013). *Dependency* parsers have seen gains from distributional statistics in the form of discrete word clusters (Koo et al., 2008), and recent work (Bansal et al., 2014) suggests that similar gains can be derived from embeddings like the ones used in this paper.

It has been less clear how (and indeed whether) word embeddings in and of themselves are useful for *constituency* parsing. There certainly exist competitive parsers that internally represent lexical items as real-valued vectors, such as the neural network-based parser of Henderson (2004), and even parsers which use pre-trained word embeddings to represent the lexicon, such as Socher et al. (2013). In these parsers, however, use of word vectors is a structural choice, rather than an added feature, and it is difficult to disentangle whether vector-space lexicons are actually more powerful than their discrete analogs—perhaps the performance of neural network parsers comes entirely from the model’s extra-lexical syntactic structure. In order to isolate the contribution from word embeddings, it is useful to demonstrate improvement over a parser that already achieves state-of-the-art performance *without* vector representations.

The fundamental question we want to explore is whether embeddings provide any information beyond what a conventional parser is able to induce from labeled parse trees. It could be that the distinctions between lexical items that embeddings capture are already modeled by parsers in other ways and therefore provide no further benefit. In this paper, we investigate this question empirically, by isolating three potential mechanisms for improvement from pre-trained word embed-

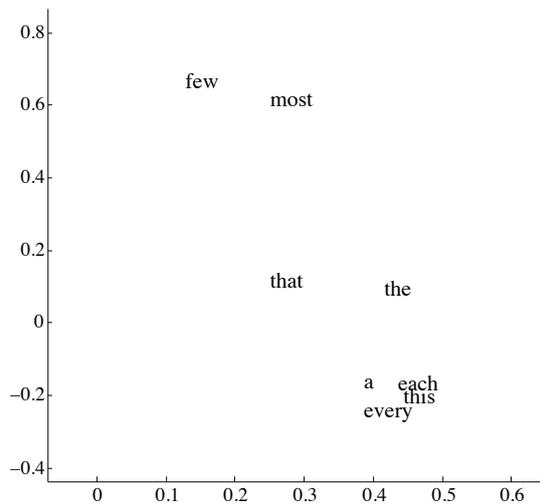


Figure 1: Word representations of English determiners, projected onto their first two principal components. Embeddings from Collobert et al. (2011).

dings. Our result is mostly negative. With extremely limited training data, parser extensions using word embeddings give modest improvements in accuracy (relative error reduction on the order of 1.5%). However, with reasonably-sized training corpora, performance does not improve even when a wide variety of embedding methods, parser modifications, and parameter settings are considered.

The fact that word embedding features result in nontrivial gains for discriminative dependency parsing (Bansal et al., 2014), but do not appear to be effective for constituency parsing, points to an interesting structural difference between the two tasks. We hypothesize that dependency parsers benefit from the introduction of features (like clusters and embeddings) that provide syntactic abstractions; but that constituency parsers already have access to such abstractions in the form of supervised preterminal tags.

2 Three possible benefits of word embeddings

We are interested in the question of whether a state-of-the-art discrete-variable constituency parser can be improved with word embeddings, and, more precisely, what aspect (or aspects) of the parser can be altered to make effective use of embeddings.

It seems clear that word embeddings exhibit some syntactic structure. Consider Figure 1,

which shows embeddings for a variety of English determiners, projected onto their first two principal components. We can see that the quantifiers *each* and *every* cluster together, as do *few* and *most*. These are precisely the kinds of distinctions between determiners that state-splitting in the Berkeley parser has shown to be useful (Petrov and Klein, 2007), and existing work (Mikolov et al., 2013b) has observed that such regular embedding structure extends to many other parts of speech. But we don’t know how prevalent or important such “syntactic axes” are in practice. Thus we have two questions: Are such groupings (learned on large data sets but from less syntactically rich models) better than the ones the parser finds on its own? How much data is needed to learn them without word embeddings?

We consider three general hypotheses about how embeddings might interact with a parser:

1. **Vocabulary expansion hypothesis:** Word embeddings are useful for handling *out-of-vocabulary* words, because they automatically ensure that unknown words are treated the same way as known words with similar representations. Example: the infrequently-occurring treebank tag UH dominates greetings (among other interjections). Upon encountering the unknown word *hey*, the parser assigns a low posterior probability of having been generated from UH. But its distributional representation is very close to the known word *hello*, and a model capable of mapping *hey* to its neighbor should be able to assign the right tag.
2. **Statistic sharing hypothesis:** Word embeddings are useful for handling *in-vocabulary* words, by making it possible to pool statistics for related words. Example: individual first names are also rare in the treebank, but tend to cluster together in distributional representations. A parser which exploited this effect could use this to acquire a robust model of name behavior by sharing statistics from all first names together, preventing low counts from producing noisy models of names.
3. **Embedding structure hypothesis:** The structure of the space used for the embeddings directly encodes syntactic information in its coordinate axes. Example: with the exception of *a*, the vertical axis in Figure 1

seems to group words by definiteness. We would expect a feature corresponding to a word’s position along this axis to be a useful feature in a feature-based lexicon.

Note that these hypotheses are not all mutually exclusive, and two or all of them might provide independent gains. Our first task is thus to design a set of orthogonal experiments which make it possible to test each of the three hypotheses in isolation. It is also possible that other mechanisms are at play that are not covered by these three hypotheses, but we consider these three to be likely central effects.

3 Parser extensions

For the experiments in this paper, we will use the Berkeley parser (Petrov and Klein, 2007) and the related Maryland parser (Huang and Harper, 2011). The Berkeley parser induces a latent, state-split PCFG in which each symbol V of the (observed) X-bar grammar is refined into a set of more specific symbols $\{V_1, V_2, \dots\}$ which capture more detailed grammatical behavior. This allows the parser to distinguish between words which share the same tag but exhibit very different syntactic behavior—for example, between articles and demonstrative pronouns. The Maryland parser builds on the state-splitting parser, replacing its basic word emission model with a feature-rich, log-linear representation of the lexicon.

The choice of this parser family has two motivations. First, these parsers are among the best in the literature, with a test performance of 90.7 F_1 for the baseline Berkeley parser on the Wall Street Journal corpus (compared to 90.4 for Socher et al. (2013) and 90.1 for Henderson (2004)). Second, and more importantly, the fact that they use no continuous state representations internally makes it easy to design experiments that isolate the contributions of word vectors, without worrying about effects from real-valued operators higher up in the model. We consider the following extensions:

Vocabulary expansion → OOV model

To evaluate the vocabulary expansion hypothesis, we introduce a simple but targeted out-of-vocabulary (OOV) model in which every unknown word is simply replaced by its nearest neighbor in the training set. For OOV words which are not in the dictionary of embeddings, we back off to the unknown word model for the underlying parser.

Statistic sharing → Lexicon pooling model

To evaluate the statistic sharing hypothesis, we propose a novel smoothing technique. The Berkeley lexicon stores, for each latent (tag, word) pair, the probability $p(w|t)$ directly in a lookup table. If we want to encourage similarly-embedded words to exhibit similar behavior in the generative model, we need to ensure that they are preferentially mapped onto the same latent preterminal tag. In order to do this, we replace this direct lookup with a smoothed, kernelized lexicon, where:

$$p(w|t) = \frac{1}{Z} \sum_{w'} \alpha_{t,w'} e^{-\beta \|\phi(w) - \phi(w')\|^2} \quad (1)$$

with Z a normalizing constant to ensure that $p(\cdot|t)$ sums to one over the entire vocabulary. $\phi(w)$ is the vector representation of the word w , $\alpha_{t,w}$ are per-basis weights, and β is an inverse radius parameter which determines the strength of the smoothing. Each $\alpha_{t,w}$ is learned in the same way as its corresponding probability in the original parser model—during each M step of the training procedure, $\alpha_{w,t}$ is set to the expected number of times the word w appears under the refined tag t . Intuitively, as β grows small groups of related words will be assigned increasingly similar probabilities of being generated from the same tag (in the limit where $\beta = 0$, Equation 1 is a uniform distribution over the entire vocabulary). As β grows large words become more independent (and in the limit where $\beta = \infty$, each summand in Equation 1 is zero except where $w' = w$, and we recover the original direct-lookup model).

There are computational concerns associated with this approach: the original scoring procedure for a (word, tag) pair was a single (constant-time) lookup; here it might take time linear in the size of the vocabulary. This causes parsing to become unacceptably slow, so an approximation is necessary. Luckily, the exponential decay of the kernel ensures that each word shares most of its weight with a small number of close neighbors, and almost none with words farther away. To exploit this, we pre-compute the k -nearest-neighbor graph of points in the embedding space, and take the sum in Equation 1 only over this set of nearest neighbors. Empirically, taking $k = 20$ gives adequate performance, and increasing it does not seem to alter the behavior of the parser.

As in the OOV model, we also need to worry about how to handle words for which we have no

vector representation. In these cases, we simply treat the words as if their vectors were so far away from everything else they had no influence, and report their weights as $p(w|t) = \alpha_w$. This ensures that our model continues to include the original Berkeley parser model as a limiting case.

Embedding structure \rightarrow embedding features

To evaluate the embedding structure hypothesis, we take the Maryland featured parser, and replace the set of lexical template features used by that parser with a set of indicator features on a discretized version of the embedding. For each dimension i , we create an indicator feature corresponding to the linearly-bucketed value of the feature at that index. In order to focus specifically on the effect of word embeddings, we remove the morphological features from the parser, but retain indicators on the identity of each lexical item.

The extensions we propose are certainly not the only way to target the hypotheses described above, but they have the advantage of being minimal and straightforwardly interpretable, and each can be reasonably expected to improve parser performance if its corresponding hypothesis is true.

4 Experimental setup

We use the Maryland implementation of the Berkeley parser as our baseline for the kernel-smoothed lexicon, and the Maryland featured parser as our baseline for the embedding-featured lexicon.¹ For all experiments, we use 50-dimensional word embeddings. Embeddings labeled C&W are from Collobert et al. (2011); embeddings labeled CBOW are from Mikolov et al. (2013a), trained with a context window of size 2.

Experiments are conducted on the Wall Street Journal portion of the English Penn Treebank. We prepare three training sets: the complete training set of 39,832 sentences from the treebank (sections 2 through 21), a smaller training set, consisting of the first 3000 sentences, and an even smaller set of the first 300.

Per-corpus-size settings of the parameter β are set by searching over several possible settings on the development set. For each training corpus size we also choose a different setting of the number of splitting iterations over which the Berkeley parser is run; for 300 sentences this is two splits, and for

¹Both downloaded from <https://code.google.com/p/umd-featured-parser/>

Model		300	3000	Full
Baseline		71.88	84.70	91.13
OOV	(C&W)	72.20	84.77	91.22
OOV	(CBOW)	72.20	84.78	91.22
Pooling	(C&W)	72.21	84.55	91.11
Pooling	(CBOW)	71.61	84.73	91.15
Features	(ident)	67.27	82.77	90.65
Features	(C&W)	70.32	83.78	91.08
Features	(CBOW)	69.87	84.46	90.86

Table 1: Contributions from OOV, lexical pooling and featured models, for two kinds of embeddings (C&W and CBOW). For both choices of embedding, the pooling and OOV models provide small gains with very little training data, but no gains on the full training set. The featured model never achieves scores higher than the generative baseline.

Model	300	3000	Full
Baseline	72.02	84.09	90.70
Pool + OOV (C&W)	72.43*	84.36*	90.11

Table 2: Test set experiments with the best combination of models (based on development experiments). Again, we observe small gains with restricted training sets but no gains on the full training set. Entries marked * are statistically significant ($p < 0.05$) under a paired bootstrap resampling test.

3000 four splits. This is necessary to avoid overfitting on smaller training sets. Consistent with the existing literature, we stop at six splits when using the full training corpus.

5 Results

Various model-specific experiments are shown in Table 1. We begin by investigating the OOV model. As can be seen, this model alone achieves small gains over the baseline for a 300-word training corpus, but these gains become statistically insignificant with more training data. This behavior is almost completely insensitive to the choice of embedding.

Next we consider the lexicon pooling model. We began by searching over exponentially-spaced values of β to determine an optimal setting for

Experiment	WSJ \rightarrow Brown	French
Baseline	86.36	74.84
Pool + OOV	86.42	75.18

Table 3: Experiments for other corpora, using the same combined model (lexicon pooling and OOV) as in Table 2. Again, we observe no significant gains over the baseline.

each training set size; as expected, for small settings of β (corresponding to aggressive smoothing) performance decreased; as we increased the parameter, performance increased slightly before tapering off to baseline parser performance. The first block in Table 1 shows the best settings of β for each corpus size; as can be seen, this also gives a small improvement on the 300-sentence training corpus, but no discernible once the system has access to a few thousand labeled sentences.

Last we consider a model with a featured lexicon, as described in Huang and Harper (2011). A baseline featured model (“ident”) contains only indicator features on word identity (and performs considerably worse than its generative counterpart on small data sets). As described above, the full featured model adds indicator features on the bucketed value of each dimension of the word embedding. Here, the trend observed in the other two models is even more prominent—embedding features lead to improvements over the featured baseline, but in no case outperform the standard baseline with a generative lexicon.

We take the best-performing combination of all of these models (based on development experiments, a combination of the lexical pooling model with $\beta = 0.3$, and OOV, both using c&w word embeddings), and evaluate this on the WSJ test set (Table 2). We observe very small (but statistically significant) gains with 300 and 3000 train sentences, but a decrease in performance on the full corpus.

To investigate the possibility that improvements from embeddings are exceptionally difficult to achieve on the Wall Street Journal corpus, or on English generally, we perform (1) a domain adaptation experiment, in which we use the OOV and lexicon pooling models to train on WSJ and test on the first 4000 sentences of the Brown corpus (the “WSJ \rightarrow Brown” column in Table 3), and (2) a multilingual experiment, in which we train and

test on the French treebank (the “French” column). Apparent gains from the OOV and lexicon pooling models remain so small as to be statistically indistinguishable.

6 Conclusion

With the goal of exploring how much useful syntactic information is provided by unsupervised word embeddings, we have presented three variations on a state-of-the-art parsing model, with extensions to the out-of-vocabulary model, lexicon, and feature set. Evaluation of these modified parsers revealed modest gains on extremely small training sets, which quickly vanish as training set size increases. Thus, at least restricted to phenomena which can be explained by the experiments described here, our results are consistent with two claims: (1) unsupervised word embeddings do contain some syntactically useful information, but (2) this information is redundant with what the model is able to determine for itself from only a small amount of labeled training data.

It is important to emphasize that these results do not argue against the use of continuous representations in a parser’s state space, nor argue more generally that constituency parsers cannot possibly benefit from word embeddings. However, the failure to uncover gains when searching across a variety of possible mechanisms for improvement, training procedures for embeddings, hyperparameter settings, tasks, and resource scenarios suggests that these gains (if they do exist) are *extremely* sensitive to these training conditions, and not nearly as accessible as they seem to be in dependency parsers. Indeed, our results suggest a hypothesis that word embeddings are useful for dependency parsing (and perhaps other tasks) because they provide a level of syntactic abstraction which is explicitly annotated in constituency parses. We leave explicit investigation of this hypothesis for future work.

Acknowledgments

This work was partially supported by BBN under DARPA contract HR0011-12-C-0014. The first author is supported by a National Science Foundation Graduate Research Fellowship.

References

- Jacob Andreas and Zoubin Ghahramani. 2013. A generative model of vector space semantics. In *Proceedings of the ACL Workshop on Continuous Vector Space Models and their Compositionality*, Sofia, Bulgaria.
- Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.
- 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.
- Scott C. Deerwester, Susan T Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.
- Dayne Freitag. 2004. Trained named entity recognition using distributional clusters. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- James Henderson. 2004. Discriminative training of a neural network statistical parser. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 95. Association for Computational Linguistics.
- Karl Moritz Hermann and Phil Blunsom. 2013. The role of syntax in vector space models of compositional semantics. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 894–904, Sofia, Bulgaria, August.
- Zhongqiang Huang and Mary P. Harper. 2011. Feature-rich log-linear lexical model for latent variable pcfg grammars. In *Proceedings of the International Joint Conference on Natural Language Processing*, pages 219–227.
- Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 595–603.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013a. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.
- Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 746–751.
- Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.
- Hinrich Schütze. 1995. Distributional part-of-speech tagging. In *Proceedings of the European Association for Computational Linguistics*, pages 141–148.
- Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing with compositional vector grammars. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Proceedings of the Annual Meeting of the Association for Computational Linguistics.

Distributed Representations of Geographically Situated Language

David Bamman Chris Dyer Noah A. Smith

School of Computer Science

Carnegie Mellon University

Pittsburgh, PA 15213, USA

{dbamman, cdyer, nasmith}@cs.cmu.edu

Abstract

We introduce a model for incorporating contextual information (such as geography) in learning vector-space representations of *situated* language. In contrast to approaches to multimodal representation learning that have used properties of the *object* being described (such as its color), our model includes information about the *subject* (i.e., the speaker), allowing us to learn the contours of a word’s meaning that are shaped by the context in which it is uttered. In a quantitative evaluation on the task of judging geographically informed semantic similarity between representations learned from 1.1 billion words of geo-located tweets, our joint model outperforms comparable independent models that learn meaning in isolation.

1 Introduction

The vast textual resources used in NLP – newswire, web text, parliamentary proceedings – can encourage a view of language as a disembodied phenomenon. The rise of social media, however, with its large volume of text paired with information about its author and social context, reminds us that each word is uttered by a particular person at a particular place and time. In short: language is *situated*.

The coupling of text with demographic information has enabled computational modeling of linguistic variation, including uncovering words and topics that are characteristic of geographical regions (Eisenstein et al., 2010; O’Connor et al., 2010; Hong et al., 2012; Doyle, 2014), learning correlations between words and socioeconomic variables (Rao et al., 2010; Eisenstein et al., 2011; Pennacchiotti and Popescu, 2011; Bamman et al., 2014); and charting how new terms spread geographically (Eisenstein et al., 2012). These models

can tell us that *hella* was (at one time) used most often by a particular demographic group in northern California, echoing earlier linguistic studies (Bucholtz, 2006), and that *wicked* is used most often in New England (Ravindranath, 2011); and they have practical applications, facilitating tasks like text-based geolocation (Wing and Baldrige, 2011; Roller et al., 2012; Ikawa et al., 2012). One desideratum that remains, however, is how the *meaning* of these terms is shaped by geographical influences – while *wicked* is used throughout the United States to mean *bad* or *evil* (“he is a wicked man”), in New England it is used as an adverbial intensifier (“my boy’s wicked smart”). In leveraging grounded social media to uncover linguistic variation, what we want to learn is how a word’s meaning is shaped by its geography.

In this paper, we introduce a method that extends vector-space lexical semantic models to learn representations of geographically situated language. Vector-space models of lexical semantics have been a popular and effective approach to learning representations of word meaning (Lin, 1998; Turney and Pantel, 2010; Reisinger and Mooney, 2010; Socher et al., 2013; Mikolov et al., 2013, *inter alia*). In bringing in extra-linguistic information to learn word representations, our work falls into the general domain of multimodal learning; while other work has used visual information to improve distributed representations (Andrews et al., 2009; Feng and Lapata, 2010; Bruni et al., 2011; Bruni et al., 2012a; Bruni et al., 2012b; Roller and im Walde, 2013), this work generally exploits information about the object being described (e.g., *strawberry* and a picture of a strawberry); in contrast, we use information about the *speaker* to learn representations that vary according to contextual variables from the speaker’s perspective. Unlike classic multimodal systems that incorporate multiple active modalities (such as gesture) from a user (Oviatt, 2003; Yu and

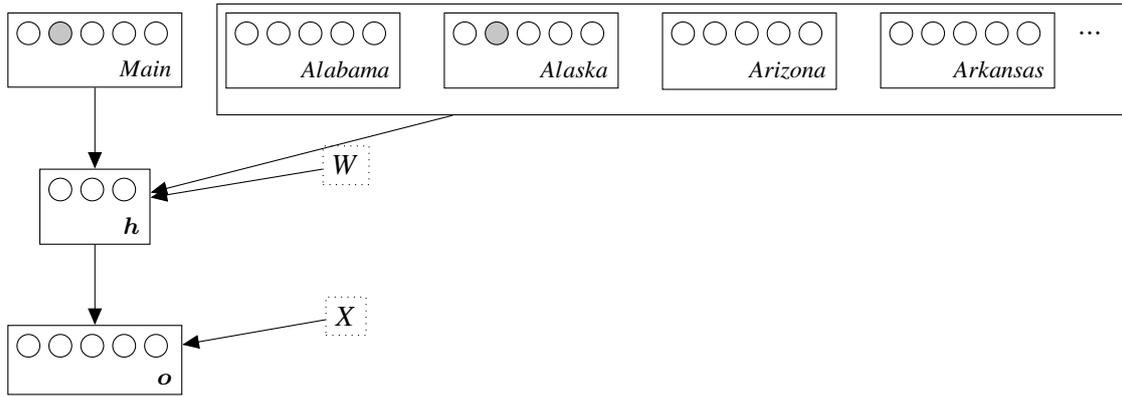


Figure 1: Model. Illustrated are the input dimensions that fire for a single sample, reflecting a particular word (vocabulary item #2) spoken in Alaska, along with a single output. Parameter matrix W consists of the learned low-dimensional embeddings.

Ballard, 2004), our primary input is textual data, supplemented with metadata about the author and the moment of authorship. This information enables learning models of word meaning that are sensitive to such factors, allowing us to distinguish, for example, between the usage of *wicked* in Massachusetts from the usage of that word elsewhere, and letting us better associate geographically grounded named entities (e.g. *Boston*) with their hypernyms (*city*) in their respective regions.

2 Model

The model we introduce is grounded in the distributional hypothesis (Harris, 1954), that two words are similar by appearing in the same kinds of contexts (where “context” itself can be variously defined as the bag or sequence of tokens around a target word, either by linear distance or dependency path). We can invoke the distributional hypothesis for many instances of regional variation by observing that such variants often appear in similar contexts. For example:

- my boy’s *wicked* smart
- my boy’s *hella* smart
- my boy’s *very* smart

Here, all three variants can often be seen in an immediately pre-adjectival position (as is common with intensifying adverbs).

Given the empirical success of vector-space representations in capturing semantic properties and their success at a variety of NLP tasks (Turian et al., 2010; Socher et al., 2011; Collobert et al., 2011; Socher et al., 2013), we use a simple, but state-of-the-art neural architecture (Mikolov et al., 2013) to learn low-dimensional real-valued repre-

sentations of words. The graphical form of this model is illustrated in figure 1.

This model corresponds to an extension of the “skip-gram” language model (Mikolov et al., 2013) (hereafter SGLM). Given an input sentence s and a context window of size t , each word s_i is conditioned on in turn to predict the identities of all of the tokens within t words around it. For a vocabulary V , each input word s_i is represented as a one-hot vector w_i of length $|V|$. The SGLM has two sets of parameters. The first is the representation matrix $W \in \mathbb{R}^{|V| \times k}$, which encodes the real-valued embeddings for each word in the vocabulary. A matrix multiply $h = w^\top W, \in \mathbb{R}^k$ serves to index the particular embedding for word w , which constitutes the model’s hidden layer. To predict the value of the context word y (again, a one-hot vector of dimensionality $|V|$), this hidden representation h is then multiplied by a second parameter matrix $X \in \mathbb{R}^{|V| \times k}$. The final prediction over the output vocabulary is then found by passing this resulting vector through the softmax function $o = \text{softmax}(Xh)$, giving a vector in the $|V|$ -dimensional unit simplex. Backpropagation using (input x , output y) word tuples learns the values of W (the embeddings) and X (the output parameter matrix) that maximize the likelihood of y (i.e., the context words) conditioned on x (i.e., the s_i ’s). During backpropagation, the errors propagated are the difference between o (a probability distribution with k outcomes) and the true (one-hot) output y .

Let us define a set of contextual variables \mathcal{C} ; in the experiments that follow, \mathcal{C} is comprised solely of geographical state $\mathcal{C}_{state} = \{AK, AL, \dots, WY\}$ but could in principle include any number of features, such as calendar

month, day of week, or other demographic variables of the speaker. Let $|\mathcal{C}|$ denote the sum of the cardinalities of all variables in \mathcal{C} (i.e., 51 states, including the District of Columbia). Rather than using a single embedding matrix W that contains low-dimensional representations for every word in the vocabulary, we define a global embedding matrix $W_{main} \in \mathbb{R}^{|V| \times k}$ and an additional $|\mathcal{C}|$ such matrices (each again of size $|V| \times k$, which capture the effect that each variable value has on each word in the vocabulary). Given an input word w and set of active variable values \mathcal{A} (e.g., $\mathcal{A} = \{state = MA\}$), we calculate the hidden layer h as the sum of these independent embeddings: $h = w^\top W_{main} + \sum_{a \in \mathcal{A}} w^\top W_a$. While the word *wicked* has a common low-dimensional representation in $W_{main, wicked}$ that is invoked for every instance of its use (regardless of the place), the corresponding vector $W_{MA, wicked}$ indicates how that common representation should shift in k -dimensional space when used in Massachusetts. Backpropagation functions as in standard SGLM, with gradient updates for each training example $\{x, y\}$ touching not only W_{main} (as in SGLM), but all active $W_{\mathcal{A}}$ as well.

The additional W embeddings we add lead to an increase in the number of total parameters by a factor of $|\mathcal{C}|$. To control for the extra degrees of freedom this entails, we add squared ℓ_2 regularization to all parameters, using stochastic gradient descent for backpropagation with minibatch updates for the regularization term. As in Mikolov et al. (2013), we speed up computation using the hierarchical softmax (Morin and Bengio, 2005) on the output matrix X .

This model defines a joint parameterization over all variable values in the data, where information from data originating in California, for instance, can influence the representations learned for Wisconsin; a naive alternative would be to simply train individual models on each variable value (a “California” model using data only from California, etc.). A joint model has three *a priori* advantages over independent models: (i) sharing data across variable values encourages representations across those values to be similar; e.g., while *city* may be closer to *Boston* in Massachusetts and *Chicago* in Illinois, in both places it still generally connotes a *municipality*; (ii) such sharing can mitigate data sparseness for less-witnessed areas; and (iii) with a joint model, all representations are guaranteed to

be in the same vector space and can therefore be compared to each other; with individual models (each with different initializations), word vectors across different states may not be directly compared.

3 Evaluation

We evaluate our model by confirming its face validity in a qualitative analysis and estimating its accuracy at the quantitative task of judging geographically-informed semantic similarity. We use 1.1 billion tokens from 93 million geolocated tweets gathered between September 1, 2011 and August 30, 2013 (approximately 127,000 tweets per day evenly sampled over those two years). This data only includes tweets that have been geolocated to state-level granularity in the United States using high-precision pattern matching on the user-specified location field (e.g., “new york ny” \rightarrow NY, “chicago” \rightarrow IL, etc.). As a pre-processing step, we identify a set of target multiword expressions in this corpus as the maximal sequence of adjectives + nouns with the highest pointwise mutual information; in all experiments described below, we define the vocabulary V as the most frequent 100,000 terms (either unigrams or multiword expressions) in the total data, and set the dimensionality of the embedding $k = 100$. In all experiments, the contextual variable is the observed US state (including DC), so that $|\mathcal{C}| = 51$; the vector space representation of word w in state s is $w^\top W_{main} + w^\top W_s$.

3.1 Qualitative Evaluation

To illustrate how the model described above can learn geographically-informed semantic representations of words, table 1 displays the terms with the highest cosine similarity to *wicked* in Kansas and Massachusetts after running our joint model on the full 1.1 billion words of Twitter data; while *wicked* in Kansas is close to other evaluative terms like *evil* and *pure* and religious terms like *gods* and *spirit*, in Massachusetts it is most similar to other intensifiers like *super*, *ridiculously* and *insanely*.

Table 2 likewise presents the terms with the highest cosine similarity to *city* in both California and New York; while the terms most evoked by *city* in California include regional locations like Chinatown, Los Angeles’ South Bay and San Francisco’s East Bay, in New York the most similar terms include *hamptons*, *upstate* and *borough*

Kansas		Massachusetts	
term	cosine	term	cosine
wicked	1.000	wicked	1.000
evil	0.884	super	0.855
pure	0.841	ridiculously	0.851
gods	0.841	insanely	0.820
mystery	0.830	extremely	0.793
spirit	0.830	goddamn	0.781
king	0.828	surprisingly	0.774
above	0.825	kinda	0.772
righteous	0.823	#sarcasm	0.772
magic	0.822	soooooo	0.770

Table 1: Terms with the highest cosine similarity to *wicked* in Kansas and Massachusetts.

California		New York	
term	cosine	term	cosine
city	1.000	city	1.000
valley	0.880	suburbs	0.866
bay	0.874	town	0.855
downtown	0.873	hamptons	0.852
chinatown	0.854	big city	0.842
south bay	0.854	borough	0.837
area	0.851	neighborhood	0.835
east bay	0.845	downtown	0.827
neighborhood	0.843	upstate	0.826
peninsula	0.840	big apple	0.825

Table 2: Terms with the highest cosine similarity to *city* in California and New York.

(New York City’s term of administrative division).

3.2 Quantitative Evaluation

As a quantitative measure of our model’s performance, we consider the task of judging semantic similarity among words whose meanings are likely to evoke strong geographical correlations. In the absence of a sizable number of linguistically interesting terms (like *wicked*) that are known to be geographically variable, we consider the proxy of estimating the named entities evoked by specific terms in different geographical regions. As noted above, geographic terms like *city* provide one such example: in Massachusetts we expect the term *city* to be more strongly connected to grounded named entities like *Boston* than to other US cities. We consider seven categories for which we can reasonably expect the connotations of each term to vary by geography; in each case, we calculate the distance between two terms x and y using representations learned for a given state ($\delta_{state}(x, y)$).

1. *city*. For each state, we measure the distance between the word *city* and the state’s most populous city; e.g., $\delta_{AZ}(city, phoenix)$.
2. *state*. For each state, the distance between

the word *state* and the state’s name; e.g., $\delta_{WI}(state, wisconsin)$.

3. *football*. For all NFL teams, the distance between the word *football* and the team name; e.g., $\delta_{IL}(football, bears)$.
4. *basketball*. For all NBA teams from a US state, the distance between the word *basketball* and the team name; e.g., $\delta_{FL}(basketball, heat)$.
5. *baseball*. For all MLB teams from a US state, the distance between the word *baseball* and the team name; e.g., $\delta_{IL}(baseball, cubs)$, $\delta_{IL}(baseball, white\ sox)$.
6. *hockey*. For all NHL teams from a US state, the distance between the word *hockey* and the team name; e.g., $\delta_{PA}(hockey, penguins)$.
7. *park*. For all US national parks, the distance between the word *park* and the park name; e.g., $\delta_{AK}(park, denali)$.

Each of these questions asks the following: what words are evoked for a given target word (like *football*)? While *football* may everywhere evoke similar sports like *baseball* or *soccer* or more specific football-related terms like *touch-down* or *field goal*, we expect that particular sports teams will be evoked more strongly by the word *football* in their particular geographical region: in Wisconsin, *football* should evoke *packers*, while in Pennsylvania, *football* evokes *steelers*. Note that this is not the same as simply asking which sports team is most frequently (or most characteristically) mentioned in a given area; by measuring the distance to a target word (*football*), we are attempting to estimate the varying strengths of association between concepts in different regions.

For each category, we measure similarity as the average cosine similarity between the vector for the target word for that category (e.g., *city*) and the corresponding vector for each state-specific answer (e.g., *chicago* for IL; *boston* for MA). We compare three different models:

1. JOINT. The full model described in section 2, in which we learn a global representation for each word along with deviations from that common representation for each state.
2. INDIVIDUAL. For comparison, we also partition the data among all 51 states, and train a single model for each state using only data from that state. In this model, there is no sharing among states; California has the most

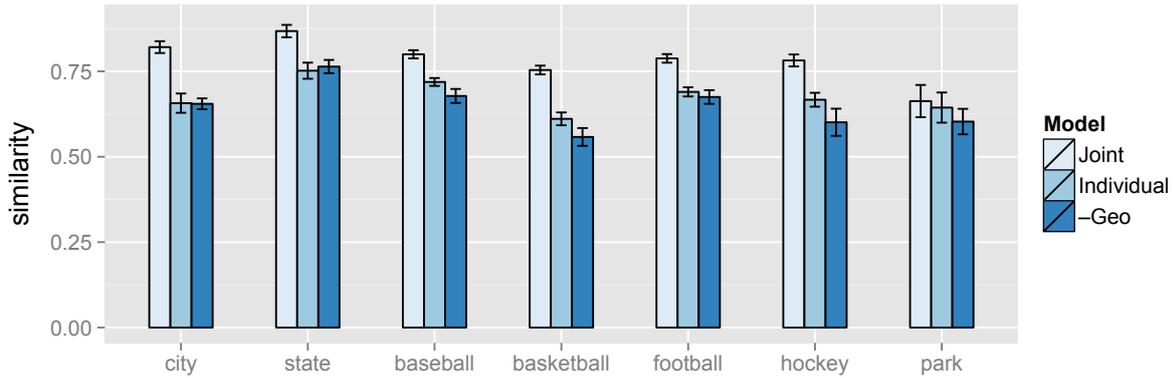


Figure 2: Average cosine similarity for all models across all categories, with 95% confidence intervals on the mean.

data with 11,604,637 tweets; Wyoming has the least with 47,503 tweets.

3. **-GEO.** We also train a single model on all of the training data, but ignore any state metadata. In this case the distance δ between two terms is their overall distance within the entire United States.

As one concrete example of these differences between individual data points, the cosine similarity between *city* and *seattle* in the **-GEO** model is 0.728 (*seattle* is ranked as the 188th most similar term to *city* overall); in the **INDIVIDUAL** model using only tweets from Washington state, $\delta_{WA}(city, seattle) = 0.780$ (rank #32); and in the **JOINT** model, using information from the entire United States with deviations for Washington, $\delta_{WA}(city, seattle) = 0.858$ (rank #6). The overall similarity for the city category of each model is the average of 51 such tests (one for each city).

Figure 2 present the results of the full evaluation, including 95% confidence intervals for each mean. While the two models that include geographical information naturally outperform the model that does not, the **JOINT** model generally far outperforms the **INDIVIDUAL** models trained on state-specific subsets of the data.¹ A model that can exploit all of the information in the data, learning core vector-space representations for all words along with deviations for each contextual variable, is able to learn more geographically-informed representations for this task than strict geographical models alone.

¹This result is robust to the choice of distance metric; an evaluation measuring the Euclidean distance between vectors shows the **JOINT** model to outperform the **INDIVIDUAL** and **-GEO** models across all seven categories.

4 Conclusion

We introduced a model for leveraging situational information in learning vector-space representations of words that are sensitive to the speaker’s social context. While our results use geographical information in learning low-dimensional representations, other contextual variables are straightforward to include as well; incorporating effects for time – such as time of day, month of year and absolute year – may be a powerful tool for revealing periodic and historical influences on lexical semantics.

Our approach explores the degree to which geography, and other contextual factors, influence word *meaning* in addition to frequency of usage. By allowing all words in different regions (or more generally, with different metadata factors) to exist in the same vector space, we are able compare different points in that space – for example, to ask what terms used in Chicago are most similar to *hot dog* in New York, or what word groups shift together in the same region in comparison to the background (indicating the shift of an entire semantic field). All datasets and software to support these geographically-informed representations can be found at: <http://www.ark.cs.cmu.edu/geoSGLM>.

5 Acknowledgments

The research reported in this article was supported by US NSF grants IIS-1251131 and CAREER IIS-1054319, and by an ARCS scholarship to D.B. This work was made possible through the use of computing resources made available by the Open Cloud Consortium, Yahoo and the Pittsburgh Supercomputing Center.

References

- Mark Andrews, Gabriella Vigliocco, and David Vinson. 2009. Integrating experiential and distributional data to learn semantic representations. *Psychological Review*, 116(3):463–498.
- David Bamman, Jacob Eisenstein, and Tyler Schnoebelen. 2014. Gender identity and lexical variation in social media. *Journal of Sociolinguistics*, 18(2).
- Elia Bruni, Giang Binh Tran, and Marco Baroni. 2011. Distributional semantics from text and images. In *Proc. of the Workshop on Geometrical Models of Natural Language Semantics*.
- Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012a. Distributional semantics in technicolor. In *Proc. of ACL*.
- Elia Bruni, Jasper Uijlings, Marco Baroni, and Nicu Sebe. 2012b. Distributional semantics with eyes: Using image analysis to improve computational representations of word meaning. In *Proc. of the ACM International Conference on Multimedia*.
- Mary Bucholtz. 2006. Word up: Social meanings of slang in California youth culture. In Jane Goodman and Leila Monaghan, editors, *A Cultural Approach to Interpersonal Communication: Essential Readings*, Malden, MA. Blackwell.
- 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.
- Gabriel Doyle. 2014. Mapping dialectal variation by querying social media. In *Proc. of EACL*.
- Jacob Eisenstein, Brendan O’Connor, Noah A. Smith, and Eric P. Xing. 2010. A latent variable model for geographic lexical variation. In *Proc. of EMNLP*.
- Jacob Eisenstein, Noah A. Smith, and Eric P. Xing. 2011. Discovering sociolinguistic associations with structured sparsity. In *Proc. of ACL*.
- Jacob Eisenstein, Brendan O’Connor, Noah A. Smith, and Eric P. Xing. 2012. Mapping the geographical diffusion of new words. *arXiv*, abs/1210.5268.
- Yansong Feng and Mirella Lapata. 2010. Visual information in semantic representation. In *Proc. of NAACL*.
- Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.
- Liangjie Hong, Amr Ahmed, Siva Gurumurthy, Alexander J. Smola, and Kostas Tsioutsoulouklis. 2012. Discovering geographical topics in the Twitter stream. In *Proc. of WWW*.
- Yohei Ikawa, Miki Enoki, and Michiaki Tatsubori. 2012. Location inference using microblog messages. In *Proc. of WWW*.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proc. of COLING-ACL*.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proc. of ICLR*.
- Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proc. of AISTATS*.
- Brendan O’Connor, Jacob Eisenstein, Eric P. Xing, and Noah A. Smith. 2010. Discovering demographic language variation. In *NIPS Workshop on Machine Learning and Social Computing*.
- Sharon Oviatt. 2003. Multimodal interfaces. In Julie A. Jacko and Andrew Sears, editors, *The Human-computer Interaction Handbook*, pages 286–304, Hillsdale, NJ, USA. L. Erlbaum Associates Inc.
- Marco Pennacchiotti and Ana-Maria Popescu. 2011. Democrats, Republicans and Starbucks aficionados: User classification in Twitter. In *Proc. of KDD*.
- Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. 2010. Classifying latent user attributes in Twitter. In *Proc. of the Workshop on Search and Mining User-generated Contents*.
- Maya Ravindranath. 2011. A wicked good reason to study intensifiers in New Hampshire. In *NWAV 40*.
- Joseph Reisinger and Raymond J. Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Proc. of NAACL*.
- Stephen Roller and Sabine Schulte im Walde. 2013. A multimodal LDA model integrating textual, cognitive and visual modalities. In *Proc. of EMNLP*.
- Stephen Roller, Michael Speriosu, Sarat Rallapalli, Benjamin Wing, and Jason Baldridge. 2012. Supervised text-based geolocation using language models on an adaptive grid. In *Proc. of EMNLP-CoNLL*.
- Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proc. of EMNLP*.
- Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. 2013. Parsing with compositional vector grammars. In *Proc. of ACL*.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proc. of ACL*.
- Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188, January.

Benjamin P. Wing and Jason Baldrige. 2011. Simple supervised document geolocation with geodesic grids. In *Proc. of ACL*.

Chen Yu and Dana H. Ballard. 2004. A multimodal learning interface for grounding spoken language in sensory perceptions. *ACM Transactions on Applied Perception*, 1(1):57–80.

Improving Multi-Modal Representations Using Image Dispersion: Why Less is Sometimes More

Douwe Kiela*, Felix Hill*, Anna Korhonen and Stephen Clark

University of Cambridge

Computer Laboratory

{douwe.kiela|felix.hill|anna.korhonen|stephen.clark}@cl.cam.ac.uk

Abstract

Models that learn semantic representations from both linguistic and perceptual input outperform text-only models in many contexts and better reflect human concept acquisition. However, experiments suggest that while the inclusion of perceptual input improves representations of certain concepts, it degrades the representations of others. We propose an unsupervised method to determine whether to include perceptual input for a concept, and show that it significantly improves the ability of multi-modal models to learn and represent word meanings. The method relies solely on image data, and can be applied to a variety of other NLP tasks.

1 Introduction

Multi-modal models that learn semantic concept representations from both linguistic and perceptual input were originally motivated by parallels with human concept acquisition, and evidence that many concepts are *grounded* in the perceptual system (Barsalou et al., 2003). Such models extract information about the perceptible characteristics of words from data collected in property norming experiments (Roller and Schulte im Walde, 2013; Silberer and Lapata, 2012) or directly from ‘raw’ data sources such as images (Feng and Lapata, 2010; Bruni et al., 2012). This input is combined with information from linguistic corpora to produce enhanced representations of concept meaning. Multi-modal models outperform language-only models on a range of tasks, including modelling conceptual association and predicting compositionality (Bruni et al., 2012; Silberer and Lapata, 2012; Roller and Schulte im Walde, 2013).

Despite these results, the advantage of multi-modal over linguistic-only models has only been

demonstrated on concrete concepts, such as *chocolate* or *cheeseburger*, as opposed to abstract concepts such as *guilt* or *obesity*. Indeed, experiments indicate that while the addition of perceptual input is generally beneficial for representations of concrete concepts (Hill et al., 2013a; Bruni et al., 2014), it can in fact be detrimental to representations of abstract concepts (Hill et al., 2013a). Further, while the theoretical importance of the perceptual modalities to concrete representations is well known, evidence suggests this is not the case for more abstract concepts (Paivio, 1990; Hill et al., 2013b). Indeed, perhaps the most influential characterization of the abstract/concrete distinction, the Dual Coding Theory (Paivio, 1990), posits that concrete representations are encoded in both the linguistic and perceptual modalities whereas abstract concepts are encoded only in the linguistic modality.

Existing multi-modal architectures generally extract and process all the information from their specified sources of perceptual input. Since perceptual data sources typically contain information about both abstract and concrete concepts, such information is included for both concept types. The potential effect of this design decision on performance is significant because the vast majority of meaning-bearing words in everyday language correspond to abstract concepts. For instance, 72% of word tokens in the British National Corpus (Leech et al., 1994) were rated by contributors to the University of South Florida dataset (USF) (Nelson et al., 2004) as more abstract than the noun *war*, a concept that many would consider quite abstract.

In light of these considerations, we propose a novel algorithm for approximating conceptual concreteness. Multi-modal models in which perceptual input is filtered according to our algorithm learn higher-quality semantic representations than previous approaches, resulting in a significant performance improvement of up to 17% in captur-

ing the semantic similarity of concepts. Further, our algorithm constitutes the first means of quantifying conceptual concreteness that does not rely on labor-intensive experimental studies or annotators. Finally, we demonstrate the application of this unsupervised concreteness metric to the semantic classification of adjective-noun pairs, an existing NLP task to which concreteness data has proved valuable previously.

2 Experimental Approach

Our experiments focus on multi-modal models that extract their perceptual input automatically from images. Image-based models more naturally mirror the process of human concept acquisition than those whose input derives from experimental datasets or expert annotation. They are also more scalable since high-quality tagged images are freely available in several web-scale image datasets.

We use *Google Images* as our image source, and extract the first n image results for each concept word. It has been shown that images from Google yield higher-quality representations than comparable sources such as *Flickr* (Bergsma and Goebel, 2011). Other potential sources, such as ImageNet (Deng et al., 2009) or the ESP Game Dataset (Von Ahn and Dabbish, 2004), either do not contain images for abstract concepts or do not contain sufficient images for the concepts in our evaluation sets.

2.1 Image Dispersion-Based Filtering

Following the motivation outlined in Section 1, we aim to distinguish visual input corresponding to concrete concepts from visual input corresponding to abstract concepts. Our algorithm is motivated by the intuition that the diversity of images returned for a particular concept depends on its concreteness (see Figure 1). Specifically, we anticipate greater congruence or similarity among a set of images for, say, *elephant* than among images for *happiness*. By exploiting this connection, the method approximates the concreteness of concepts, and provides a basis to filter the corresponding perceptual information.

Formally, we propose a measure, *image dispersion* d of a concept word w , defined as the average pairwise cosine distance between all the image representations $\{\vec{w}_1 \dots \vec{w}_n\}$ in the set of images for that concept:



Figure 1: Example images for a concrete (*elephant* – little diversity, low dispersion) and an abstract concept (*happiness* – greater diversity, high dispersion).

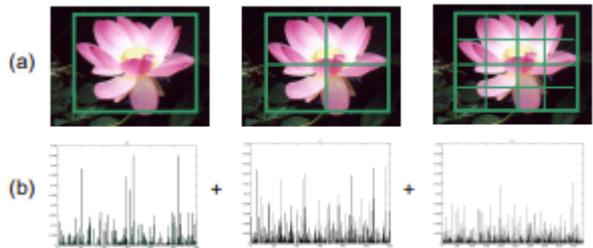


Figure 2: Computation of PHOW descriptors using dense SIFT for levels $l = 0$ to $l = 2$ and the corresponding histogram representations (Bosch et al., 2007).

$$d(w) = \frac{1}{2n(n-1)} \sum_{i < j \leq n} 1 - \frac{\vec{w}_i \cdot \vec{w}_j}{|\vec{w}_i| |\vec{w}_j|} \quad (1)$$

We use an average pairwise distance-based metric because this emphasizes the total variation more than e.g. the mean distance from the centroid. In all experiments we set $n = 50$.

Generating Visual Representations Visual vector representations for each image were obtained using the well-known *bag of visual words* (BoVW) approach (Sivic and Zisserman, 2003). BoVW obtains a vector representation for an

image by mapping each of its local descriptors to a cluster histogram using a standard clustering algorithm such as k-means.

Previous NLP-related work uses *SIFT* (Feng and Lapata, 2010; Bruni et al., 2012) or *SURF* (Roller and Schulte im Walde, 2013) descriptors for identifying points of interest in an image, quantified by 128-dimensional local descriptors. We apply *Pyramid Histogram Of visual Words* (PHOW) descriptors, which are particularly well-suited for object categorization, a key component of image similarity and thus dispersion (Bosch et al., 2007). PHOW is roughly equivalent to running SIFT on a dense grid of locations at a fixed scale and orientation and at multiple scales (see Fig 2), but is both more efficient and more accurate than regular (dense) SIFT approaches (Bosch et al., 2007). We resize the images in our dataset to 100x100 pixels and compute PHOW descriptors using *VLFeat* (Vedaldi and Fulkerson, 2008).

The descriptors for the images were subsequently clustered using mini-batch *k*-means (Sculley, 2010) with $k = 50$ to obtain histograms of visual words, yielding 50-dimensional visual vectors for each of the images.

Generating Linguistic Representations We extract continuous vector representations (also of 50 dimensions) for concepts using the continuous log-linear skipgram model of Mikolov et al. (2013a), trained on the 100M word British National Corpus (Leech et al., 1994). This model learns high quality lexical semantic representations based on the distributional properties of words in text, and has been shown to outperform simple distributional models on applications such as semantic composition and analogical mapping (Mikolov et al., 2013b).

2.2 Evaluation Gold-standards

We evaluate models by measuring the Spearman correlation of model output with two well-known gold-standards reflecting semantic proximity – a standard measure for evaluating the quality of representations (see e.g. Agirre et al. (2009)).

To test the ability of our model to capture concept similarity, we measure correlations with WordSim353 (Finkelstein et al., 2001), a selection of 353 concept pairs together with a similarity rating provided by human annotators. WordSim has been used as a benchmark for distributional semantic models in numerous studies (see

e.g. (Huang et al., 2012; Bruni et al., 2012)).

As a complementary gold-standard, we use the University of South Florida Norms (USF) (Nelson et al., 2004). This dataset contains scores for *free association*, an experimental measure of cognitive association, between over 40,000 concept pairs. The USF norms have been used in many previous studies to evaluate semantic representations (Andrews et al., 2009; Feng and Lapata, 2010; Silberer and Lapata, 2012; Roller and Schulte im Walde, 2013). The USF evaluation set is particularly appropriate in the present context because concepts in the dataset are also rated for conceptual concreteness by at least 10 human annotators.

We create a representative evaluation set of USF pairs as follows. We randomly sample 100 concepts from the upper quartile and 100 concepts from the lower quartile of a list of all USF concepts ranked by concreteness. We denote these sets C , for *concrete*, and A for abstract respectively. We then extract all pairs (w_1, w_2) in the USF dataset such that both w_1 and w_2 are in $A \cup C$. This yields an evaluation set of 903 pairs, of which 304 are such that $w_1, w_2 \in C$ and 317 are such that $w_1, w_2 \in A$.

The images used in our experiments and the evaluation gold-standards can be downloaded from <http://www.cl.cam.ac.uk/~dk427/dispersion.html>.

3 Improving Multi-Modal Representations

We apply *image dispersion-based filtering* as follows: if both concepts in an evaluation pair have an image dispersion below a given threshold, both the linguistic and the visual representations are included. If not, in accordance with the Dual Coding Theory of human concept processing (Paivio, 1990), only the linguistic representation is used. For both datasets, we set the threshold as the median image dispersion, although performance could in principle be improved by adjusting this parameter. We compare dispersion filtered representations with linguistic, perceptual and standard multi-modal representations (concatenated linguistic and perceptual representations). Similarity between concept pairs is calculated using cosine similarity.

As Figure 3 shows, dispersion-filtered multi-modal representations significantly outperform

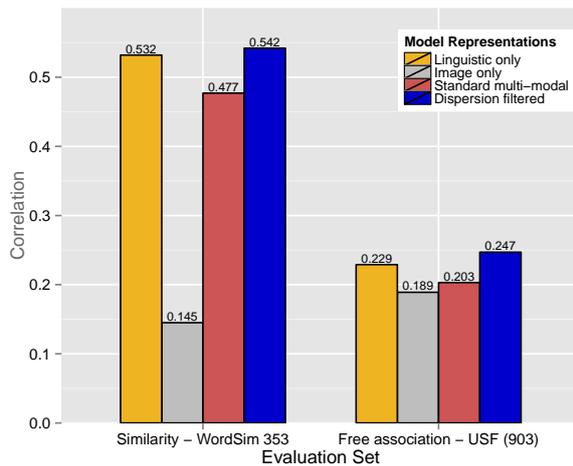


Figure 3: Performance of conventional multi-modal (visual input included for all concepts) vs. image dispersion-based filtering models (visual input only for concepts classified as concrete) on the two evaluation gold-standards.

standard multi-modal representations on both evaluation datasets. We observe a 17% increase in Spearman correlation on WordSim353 and a 22% increase on the USF norms. Based on the correlation comparison method of Steiger (1980), both represent significant improvements (WordSim353, $t = 2.42, p < 0.05$; USF, $t = 1.86, p < 0.1$). In both cases, models with the dispersion-based filter also outperform the purely linguistic model, which is not the case for other multi-modal approaches that evaluate on WordSim353 (e.g. Bruni et al. (2012)).

4 Concreteness and Image Dispersion

The filtering approach described thus far improves multi-modal representations because image dispersion provides a means to distinguish concrete concepts from more abstract concepts. Since research has demonstrated the applicability of concreteness to a range of other NLP tasks (Turney et al., 2011; Kwong, 2008), it is important to examine the connection between image dispersion and concreteness in more detail.

4.1 Quantifying Concreteness

To evaluate the effectiveness of image dispersion as a proxy for concreteness we evaluated our algorithm on a binary classification task based on the set of 100 concrete and 100 abstract concepts AUC introduced in Section 2. By classifying con-

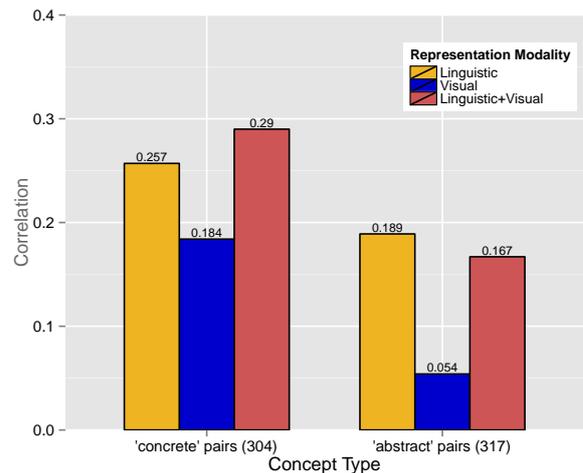


Figure 4: Visual input is valuable for representing concepts that are classified as concrete by the image dispersion algorithm, but not so for concepts classified as abstract. All correlations are with the USF gold-standard.

cepts with image dispersion below the median as concrete and concepts above this threshold as abstract we achieved an abstract-concrete prediction accuracy of 81%.

While well-understood intuitively, concreteness is not a formally defined notion. Quantities such as the USF concreteness score depend on the subjective judgement of raters and the particular annotation guidelines. According to the Dual Coding Theory, however, concrete concepts are precisely those with a salient perceptual representation. As illustrated in Figure 4, our binary classification conforms to this characterization. The importance of the visual modality is significantly greater when evaluating on pairs for which both concepts are classified as concrete than on pairs of two abstract concepts.

Image dispersion is also an effective predictor of concreteness on samples for which the abstract/concrete distinction is less clear. On a different set of 200 concepts extracted by random sampling from the USF dataset stratified by concreteness rating (including concepts across the concreteness spectrum), we observed a high correlation between abstractness and dispersion (Spearman $\rho = 0.61, p < 0.001$). On this more diverse sample, which reflects the range of concepts typically found in linguistic corpora, image dispersion is a particularly useful diagnostic for identifying

Concept	Image Dispersion	Conc. (USF)
<i>shirt</i>	.488	6.05
<i>bed</i>	.495	5.91
<i>knife</i>	.560	6.08
<i>dress</i>	.578	6.59
<i>car</i>	.580	6.35
<i>ego</i>	1.000	1.93
<i>nonsense</i>	.999	1.90
<i>memory</i>	.999	1.78
<i>potential</i>	.997	1.90
<i>know</i>	.996	2.70

Table 1: Concepts with highest and lowest image dispersion scores in our evaluation set, and concreteness ratings from the USF dataset.

the very abstract or very concrete concepts. As Table 1 illustrates, the concepts with the lowest dispersion in this sample are, without exception, highly concrete, and the concepts of highest dispersion are clearly very abstract.

It should be noted that all previous approaches to the automatic measurement of concreteness rely on annotator ratings, dictionaries or manually-constructed resources. Kwong (2008) proposes a method based on the presence of hard-coded phrasal features in dictionary entries corresponding to each concept. By contrast, Sánchez et al. (2011) present an approach based on the position of word senses corresponding to each concept in the WordNet ontology (Fellbaum, 1999). Turney et al. (2011) propose a method that extends a large set of concreteness ratings similar to those in the USF dataset. The Turney et al. algorithm quantifies the concreteness of concepts that lack such a rating based on their proximity to rated concepts in a semantic vector space. In contrast to each of these approaches, the image dispersion approach requires no hand-coded resources. It is therefore more scalable, and instantly applicable to a wide range of languages.

4.2 Classifying Adjective-Noun Pairs

Finally, we explored whether image dispersion can be applied to specific NLP tasks as an effective proxy for concreteness. Turney et al. (2011) showed that concreteness is applicable to the classification of adjective-noun modification as either literal or non-literal. By applying a logistic regression with noun concreteness as the predictor variable, Turney et al. achieved a classification accu-

racy of 79% on this task. This model relies on significant supervision in the form of over 4,000 human lexical concreteness ratings.¹ Applying image dispersion in place of concreteness in an identical classifier on the same dataset, our entirely unsupervised approach achieves an accuracy of 63%. This is a notable improvement on the largest-class baseline of 55%.

5 Conclusions

We presented a novel method, image dispersion-based filtering, that improves multi-modal representations by approximating conceptual concreteness from images and filtering model input. The results clearly show that including more perceptual input in multi-modal models is not always better. Motivated by this fact, our approach provides an intuitive and straightforward metric to determine whether or not to include such information.

In addition to improving multi-modal representations, we have shown the applicability of the image dispersion metric to several other tasks. To our knowledge, our algorithm constitutes the first unsupervised method for quantifying conceptual concreteness as applied to NLP, although it does, of course, rely on the Google Images retrieval algorithm. Moreover, we presented a method to classify adjective-noun pairs according to modification type that exploits the link between image dispersion and concreteness. It is striking that this apparently linguistic problem can be addressed solely using the raw data encoded in images.

In future work, we will investigate the precise quantity of perceptual information to be included for best performance, as well as the optimal filtering threshold. In addition, we will explore whether the application of image data, and the interaction between images and language, can yield improvements on other tasks in semantic processing and representation.

Acknowledgments

DK is supported by EPSRC grant EP/I037512/1. FH is supported by St John’s College, Cambridge. AK is supported by The Royal Society. SC is supported by ERC Starting Grant DisCoTex (306920) and EPSRC grant EP/I037512/1. We thank the anonymous reviewers for their helpful comments.

¹The MRC Psycholinguistics concreteness ratings (Coltheart, 1981) used by Turney et al. (2011) are a subset of those included in the USF dataset.

References

- Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 19–27, Boulder, Colorado.
- Mark Andrews, Gabriella Vigliocco, and David Vinson. 2009. Integrating experiential and distributional data to learn semantic representations. *Psychological review*, 116(3):463.
- Lawrence W Barsalou, W Kyle Simmons, Aron K Barbey, and Christine D Wilson. 2003. Grounding conceptual knowledge in modality-specific systems. *Trends in cognitive sciences*, 7(2):84–91.
- Shane Bergsma and Randy Goebel. 2011. Using visual information to predict lexical preference. In *RANLP*, pages 399–405.
- Anna Bosch, Andrew Zisserman, and Xavier Munoz. 2007. Image classification using random forests and ferns. In *Proceedings of ICCV*.
- Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 136–145. Association for Computational Linguistics.
- Elia Bruni, Nam Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, 49:1–47.
- Max Coltheart. 1981. The MRC psycholinguistic database. *The Quarterly Journal of Experimental Psychology*, 33(4):497–505.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.
- Christiane Fellbaum. 1999. *WordNet*. Wiley Online Library.
- Yansong Feng and Mirella Lapata. 2010. Visual information in semantic representation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 91–99. Association for Computational Linguistics.
- Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppín. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM.
- Felix Hill, Douwe Kiela, and Anna Korhonen. 2013a. Concreteness and corpora: A theoretical and practical analysis. *CMCL 2013*.
- Felix Hill, Anna Korhonen, and Christian Bentz. 2013b. A quantitative empirical analysis of the abstract/concrete distinction. *Cognitive science*, 38(1).
- Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics.
- Oi Yee Kwong. 2008. A preliminary study on the impact of lexical concreteness on word sense disambiguation. In *PACLIC*, pages 235–244.
- Geoffrey Leech, Roger Garside, and Michael Bryant. 1994. Claws4: the tagging of the british national corpus. In *Proceedings of the 15th conference on Computational linguistics-Volume 1*, pages 622–628. Association for Computational Linguistics.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of International Conference of Learning Representations*, Scottsdale, Arizona, USA.
- 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.
- Douglas L Nelson, Cathy L McEvoy, and Thomas A Schreiber. 2004. The University of South Florida free association, rhyme, and word fragment norms. *Behavior Research Methods, Instruments, & Computers*, 36(3):402–407.
- Allan Paivio. 1990. *Mental representations: A dual coding approach*. Oxford University Press.
- Stephen Roller and Sabine Schulte im Walde. 2013. A multimodal LDA model integrating textual, cognitive and visual modalities. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1146–1157, Seattle, Washington, USA, October. Association for Computational Linguistics.
- David Sánchez, Montserrat Batet, and David Isern. 2011. Ontology-based information content computation. *Knowledge-Based Systems*, 24(2):297–303.
- D Sculley. 2010. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178. ACM.
- Carina Silberer and Mirella Lapata. 2012. Grounded models of semantic representation. In *Proceedings of the 2012 Joint Conference on Empirical Methods*

in Natural Language Processing and Computational Natural Language Learning, pages 1423–1433. Association for Computational Linguistics.

J. Sivic and A. Zisserman. 2003. Video Google: a text retrieval approach to object matching in videos. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, volume 2, pages 1470–1477, Oct.

James H Steiger. 1980. Tests for comparing elements of a correlation matrix. *Psychological Bulletin*, 87(2):245.

Peter D Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the 2011 Conference on the Empirical Methods in Natural Language Processing*, pages 680–690.

A. Vedaldi and B. Fulkerson. 2008. VLFeat: An open and portable library of computer vision algorithms. <http://www.vlfeat.org/>.

Luis Von Ahn and Laura Dabbish. 2004. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM.

Bilingual Event Extraction: a Case Study on Trigger Type Determination

Zhu Zhu[†] Shoushan Li^{†*} Guodong Zhou[†] Rui Xia[‡]

[†]Natural Language Processing Lab
Soochow University, China
{zhuzhu0020,
shoushan.li}@gmail.com,
gdzhou@suda.edu.cn

[‡]Department of Computer Science
Nanjing University of Science and
Technology
rxia@njjust.edu.cn

Abstract

Event extraction generally suffers from the data sparseness problem. In this paper, we address this problem by utilizing the labeled data from two different languages. As a preliminary study, we mainly focus on the sub-task of trigger type determination in event extraction. To make the training data in different languages help each other, we propose a uniform text representation with bilingual features to represent the samples and handle the difficulty of locating the triggers in the translated text from both monolingual and bilingual perspectives. Empirical studies demonstrate the effectiveness of the proposed approach to bilingual classification on trigger type determination.

1 Introduction

Event extraction is an increasingly hot and challenging research topic in the natural language processing (NLP) community (Ahn, 2006; Saun et al. 2006; Zhao et al. 2008). It aims to automatically extract certain types of events with the arguments to present the texts under a structured form. In event extraction, there are four primary subtasks, named trigger identification, trigger type determination, argument identification, and argument role determination (Chen and NG, 2012). As an important technology in information extraction, event extraction could be applied to many fields such as information retrieval, summarization, text mining, and question answering.

Recently, the dominative approach to event extraction is based on supervised learning where a set of labeled samples are exploited to train a model to extract the events. However, the availa-

ble labeled data are rather sparse due to various kinds of event categories. For example, the event taxonomy in ACE 2005¹ (Automatic Content Extraction) includes 8 types of events, with 33 subtypes, such as “*Marry/Life*” (subtype/type), and “*Transport/Movement*”. Moreover, some subtypes such as “*Nominate/Personnel*” and “*Convict/Justice*” contain less than 10 labeled samples in the English and Chinese corpus respectively. Apparently, such a small scale of training data is difficult to yield a satisfying performance.

One possible way to alleviate the data sparseness problem in event extraction is to conduct bilingual event extraction with training data from two different languages. This is motivated by the fact that labeled data from a language is highly possible to convey similar information in another language. For example, **E1** is an event sample from the English corpus and **E2** is another one in the Chinese corpus. Apparently, **E1** and the English translation text of **E2**, share some important clues such as *meet* and *Iraq* which highly indicates the event type of “*Meet/Contact*”.

E1: *Bush arrived in Saint Petersburg on Saturday, when he also briefly met German chancellor Gerhard Schroeder, whose opposition to the Iraq war had soured his relationship with Washington, at a dinner hosted by Putin.*

E2: *美国总统布什将于2月访问德国并与施罗德会谈，伊朗和伊拉克问题将是双方会谈的重点。(U.S. president George W. Bush will visit Germany in February and meet with Schroeder, Iran and Iraq will be the focus of the talks the two sides.)*

In this paper, we address the data sparseness problem in event extraction with a bilingual pro-

* Corresponding author

¹<http://www.nist.gov/speech/tests/ace/2005>

cessing approach which aims to exploit bilingual training data to enhance the extraction performance in each language. As a preliminary work, we mainly focus on the subtask of trigger type determination. Accordingly, our goal is to design a classifier which is trained with labeled data from two different languages and is capable of classifying the test data from both languages. Generally, this task possesses two main challenges.

The first challenge is text representation, namely, how to eliminate the language gap between the two languages. To tackle this, we first employ Google Translate², a state-of-the-art machine translation system, to gain the translation of an event instance, similar to what has been widely done by previous studies in bilingual classification tasks e.g., Wan (2008); Then, we uniformly represent each text with bilingual word features. That is, we augment each original feature vector into a novel one which contains the translated features.

The second challenge is the translation for some specific features. It is well-known that some specific features, such as the triggers and their context features, are extremely important for determining the event types. For example, in **E3**, both trigger “*left*” and named entity “*Saddam*” are important features to tell the event type, i.e., “*Transport/Movement*”. When it is translated to Chinese, it is also required to know trigger “*离开*”(left) and named entity “*萨达姆*”(Saddam) in **E4**, the Chinese translation of **E3**.

E3: *Saddam's clan is said to have left for a small village in the desert.*

E4: Chinese translation: 据说 萨达姆 (Saddam) 家族已经 离开(left) 沙漠中的一个小村庄。

However, it is normally difficult to know which words are the triggers and surrounding entities in the translated sentence. To tackle this issue, we propose to locate the trigger from both monolingual and bilingual perspectives in the translation text. Empirical studies demonstrate that adding the translation of these specific features substantially improves the classification performance.

The remainder of this paper is organized as follows. Section 2 overviews the related work on event extraction. Section 3 proposes our ap-

proach to bilingual event extraction. Section 4 gives the experimental studies. In Section 5, we conclude our work and give some future work.

2 Related Work

In the NLP community, event extraction has been mainly studied in both English and Chinese.

In English, various supervised learning approaches have been explored recently. Bethard and Martin (2006) formulate the event identification as a classification problem in a word-chunking paradigm, introducing a variety of linguistically motivated features. Ahn (2006) proposes a trigger-based method. It first identifies the trigger in an event, and then uses a multi-classifier to implement trigger type determination. Ji and Grishman (2008) employ an approach to propagate consistent event arguments across sentences and documents. Liao and Grishman (2010) apply document level information to improve the performance of event extraction. Hong et al. (2011) leverage cross-entity information to improve traditional event extraction, regarding entity type consistency as a key feature. More recently, Li et al. (2013) propose a joint framework based on structured prediction which extracts triggers and arguments together.

In Chinese, relevant studies in event extraction are in a relatively primary stage with focus on more special characteristics and challenges. Tan et al. (2008) employ local feature selection and explicit discrimination of positive and negative features to ensure the performance of trigger type determination. Chen and Ji (2009) apply lexical, syntactic and semantic features in trigger labeling and argument labeling to improve the performance. More recently, Li et al. (2012) and Li et al. (2013) introduce two inference mechanisms to infer unknown triggers and recover trigger mentions respectively with morphological structures.

In comparison with above studies, we focus on bilingual event extraction. Although bilingual classification has been paid lots of attention in other fields (Wan 2008; Haghghi et al., 2008; Ismail et al., 2010; Lu et al., 2011; Li et al., 2013), there is few related work in event extraction. The only one related work we find is Ji (2009) which proposes an inductive learning approach to exploit cross-lingual predicate clusters to improve the event extraction task with the main goal to get the event taggers from extra resources, i.e., an English and Chinese parallel corpus. Differently, our goal is to make the la-

² www.google.com

beled data from two languages help each other without any other extra resources, which is original in the study of event extraction.

3 The Proposed Approach

Trigger type determination aims to determine the event type of a trigger given the trigger and its context (e.g., a sentence). Existing approaches to trigger type determination mainly focus on monolingual classification. Figure 1 illustrates the framework for Chinese and English.

In comparison, our approach exploits the corpora from two different languages. Figure 2 illustrates the framework. As shown in the figure, we first get the translated corpora of Chinese and English origin corpora through machine translation. Then, we represent each text with bilingual features, which enables us to merge the training data from both languages so as to make them help each other.

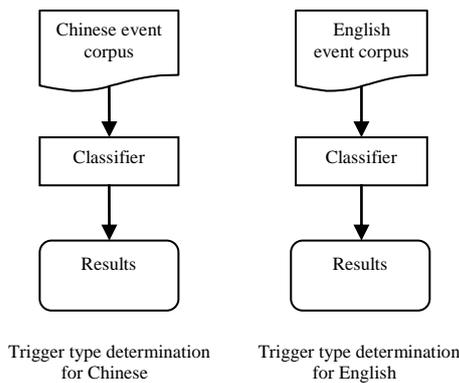


Figure 1: The framework of monolingual classification for trigger type determination

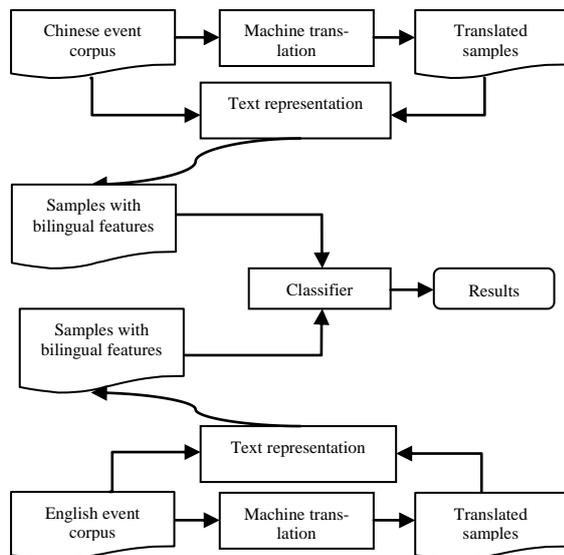


Figure 2: The framework of bilingual classification for trigger type determination

3.1 Text Representation

In a supervised learning approach, labeled data is trained to obtain a classifier. In this approach, the extracted features are the key components to make a successful classifier. Table 1 shows some typical kinds of features in a monolingual classification task for trigger type determination. To better understand these features, the real feature examples in **E3** are given in the table.

Given the feature definition, a monolingual sample x is represented as the combination of all the features, i.e.,

$$x = \left(e_1, e_2, \dots, e_n, Tri, POS_Tri, Tri_con, POS_con, Ent, Ent_type, Ent_subtype \right) \quad (1)$$

Features	Feature examples in E3
All words (e_1, e_2, \dots, e_n)	<i>Saddam, clan, is, ... , desert</i>
Trigger (Tri)	<i>left</i>
POS of the trigger (POS_Tri)	<i>VBN</i>
Trigger's context words (Tri_con)	<i>...,have, for,...</i>
POS of trigger's context words (POS_con)	<i>...,VB,IN,...</i>
Entities around trigger (Ent)	<i>Saddam</i>
Entity type (Ent_type)	<i>PER</i>
Entity subtype ($Ent_subtype$)	<i>individual</i>

Table 1: The features and some feature examples for trigger type determination

In bilingual classification, we represent a sample with bilingual features, which makes it possible to train with the data from two languages. To achieve this goal, we employ a single feature augmentation strategy to augment the monolingual features into bilingual features, i.e.,

$$x \Rightarrow x_{Chinese}, x_{English} \quad (2)$$

Specifically, a sample x is represented as follows:

$$x = \left(\begin{array}{l} c_1, c_2, \dots, c_m, Tri_c, POS_Tri_c, Tri_c_con, \\ POS_con, Ent_c, Ent_type, Ent_subtype \\ e_1, e_2, \dots, e_n, Tri_e, POS_Tri_e, Tri_e_con, \\ POS_con, Ent_e, Ent_type, Ent_subtype \end{array} \right) \quad (3)$$

Where the tokens with the 'c'/'e' subscript mean the features generated from the Chinese/English text. From the features, we can see that some

features, such as Tri_{con} and Ent , depend on the location of the trigger word. Therefore, locating the trigger in the translated text becomes crucial.

3.2 Locating Translated Trigger

Without loss of generality, we consider the case of translating a Chinese event sample into an English one. Formally, the word sequence of a Chinese event sample is denoted as $s_c = (c_1, c_2, \dots, c_n)$, while the sequence of the translated one is denoted as $s_e = (e_1, e_2, \dots, e_m)$. Then, the objective is to get the English trigger Tri_e in s_e , given the Chinese trigger word Tri_c in s_c . The objective function is given as follows:

$$\arg \max_{1 \leq k, l \leq m} P(e_{k-l} = Tri_e) \quad (4)$$

Where e_{k-l} denotes the substring $(e_k, e_{k+1}, \dots, e_l)$ in s_e and $1 \leq k, l \leq m$.

In this paper, the above function could be solved in two perspectives: monolingual and bilingual ones. The former uses the English training data alone to locate the trigger while the latter exploit the bilingual information to get the translated counterpart of the Chinese trigger.

The monolingual perspective: The objective is to locate the trigger with the monolingual information. That is,

$$\arg \max_{1 \leq k, l \leq m} P(e_{k-l} = Tri_e | s_e, R_e) \quad (5)$$

Where R_e denotes the training resource in English. In fact, this task is exactly the first subtask in event extraction named trigger identification, as mentioned in Introduction. For a simplified implementation, we first estimate the probabilities of $P(e_{k-l} = Tri_e)$ in R_e with maximum likelihood estimation when $e_{k-l} \in s_e$.

The bilingual perspective: The objective is to locate the trigger with the bilingual information. That is,

$$\arg \max_{1 \leq k, l \leq m} P(e_{k-l} = Tri_e | s_e, s_c, Tri_c) \quad (6)$$

Where Tri_c is the trigger word in Chinese and s_e is the translated text towards s_c . More generally, this can be solved from a standard word alignment model in machine translation (Och et al, 1999; Koehn et al, 2003). However, training a

word alignment requires a huge parallel corpus which is not available here.

For a simplified implementation, we first get the Tri_c 's translation, denoted as $trans_{Tri_c}$, with Google Translate. Then, we estimate $P(e_{k-l} = Tri_e)$ as follows:

$$P(e_{k-l} = Tri_e) = \begin{cases} 0.9 & \text{if } e_{k-l} = trans_{Tri_c} \\ \alpha & \text{others} \end{cases} \quad (7)$$

Where 0.9 is an empirical value which makes the translation probability become a dominative factor when the translation of the trigger is found in the translated sentence. α is a small value which makes the sum of all probabilities equals 1.

The final decision is made according to both the monolingual and bilingual perspectives, i.e.,

$$\arg \max_{1 \leq k, l \leq m} P(e_{k-l} = Tri_e | s_e, R_e) \cdot P(e_{k-l} = Tri_e | s_e, s_c, Tri_c) \quad (8)$$

Note that we reduce the computational cost by make the word length of the trigger less than 3, i.e., $l - k \leq 3$.

4 Experimentation

4.1 Experimental Setting

Data sets: The Chinese and English corpus for event extraction are from ACE2005, which involves 8 types and 33 subtypes. All our experiments are conducted on the subtype case. Due to the space limit, we only report the statistics for each type, as shown in Table 2. For each subtype, 80% samples are used as training data while the rest are as test data.

#	Chinese	English	total
Life	389	902	1291
Movement	593	679	1272
Transaction	147	379	526
Business	144	137	281
Conflict	514	1629	2143
Contact	263	373	636
Personnel	203	514	717
Justice	457	672	1129
total	2710	5285	7995

Table 2: Statistics in each event type in both Chinese and English data sets

Features: The features have been illustrated in Table 1 in Section 3.2.

Classification algorithm: The maximum entropy (ME) classifier is implemented with the public tool, Mallet Toolkits³.

Evaluation metric: The performance of event type recognition is evaluated with F-score.

4.2 Experimental Results

In this section, we evaluate the performance of our approach to bilingual classification on trigger type determination. For comparison, following approaches are implemented:

- **Monolingual:** perform monolingual classification on the Chinese and English corpus individually, as shown in Figure 1.
- **Bilingual:** perform bilingual classification with partial bilingual features, ignoring the context features (e.g., context words, context entities) under the assumption that the trigger location task is not done.
- **Bilingual_location:** perform bilingual classification by translating each sample into another language and using a uniform representation with all bilingual features as shown in Section 3.2. This is exactly our approach. The number of the context words and entities before or after the trigger words is set as 3.

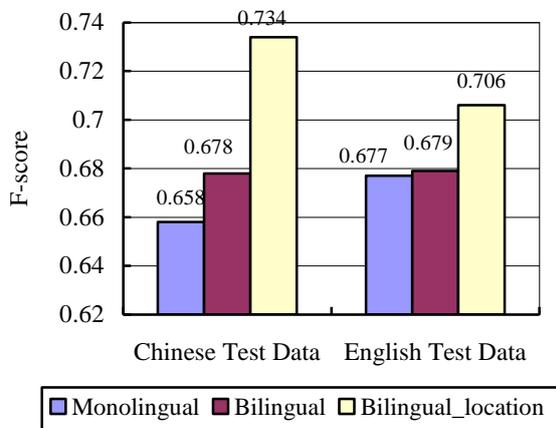


Figure 3: Performance comparison of the three approaches on the Chinese and English test data

Figure 3 shows the classification results of the three approaches on the Chinese and English test data. From this figure, we can see that **Bilingual_location** apparently outperform **Monolingual**, which verifies the effectiveness of using bilingual corpus. Specifically, the improvement by our approach in Chinese is impressive, reaching 7.6%. The results also demonstrate the importance of the operation of the trigger location,

without which, bilingual classification can only slightly improve the performance, as shown in the English test data.

The results demonstrate that our bilingual classification approaches are more effective for the Chinese data. This is understandable because the size of English data is much larger than that of Chinese data, 5285 vs. 2710, as shown in Table 2. Specifically, after checking the results in each subtype, we find that some subtypes in Chinese have very few samples while corresponding subtypes in English have a certain number samples. For example, the subtype of “*Elect/Personnel*” only contains 30 samples in the Chinese data while 161 samples can be found in the English data, which leads a very high improvement (15.4%) for the Chinese test data. In summary, our bilingual classification approach provides an effective way to handle the data sparseness problem in event extraction.

5 Conclusion and Future Work

This paper addresses the data sparseness problem in event extraction by proposing a bilingual classification approach. In this approach, we use a uniform text representation with bilingual features and merge the training samples from both languages to enlarge the size of the labeled data. Furthermore, we handle the difficulty of locating the trigger from both the monolingual and bilingual perspectives. Empirical studies show that our approach is effective in using bilingual corpus to improve monolingual classification in trigger type determination.

Bilingual event extraction is still in its early stage and many related research issues need to be investigated in the future work. For example, it is required to propose novel approaches to the bilingual processing tasks in other subtasks of event extraction. Moreover, it is rather challenging to consider a whole bilingual processing framework when all these subtasks are involved together.

Acknowledgments

This research work has been partially supported by two NSFC grants, No.61375073, and No.61273320, one National High-tech Research and Development Program of China No.2012AA011102, one General Research Fund (GRF) project No.543810 and one Early Career Scheme (ECS) project No.559313 sponsored by the Research Grants Council of Hong Kong, the NSF grant of Zhejiang Province No.Z1110551.

³ <http://mallet.cs.umass.edu/>

References

- Ahn D. 2006. The Stages of Event Extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pp.1~8.
- Bethard S. and J. Martin. 2006. Identification of Event Mentions and Their Semantic Class. In *Proceedings of EMNLP-2006*, pp.146-154.
- Chen C. and V. NG. 2012. Joint Modeling for Chinese Event Extraction with Rich Linguistic Features. In *Proceedings of COLING-2012*, pp. 529-544.
- Chen Z. and H. Ji. 2009. Language Specific Issue and Feature Exploration in Chinese Event Extraction. In *Proceedings of NAACL-2009*, pp. 209-212.
- Haghighi A., P. Liang, T. Berg-Kirkpatrick and D. Klein. 2008. Learning Bilingual Lexicons from Monolingual Corpora. In *Proceedings of ACL-2008*, pp. 771-779.
- Hong Y., J. Zhang., B. Ma., J. Yao., and G. Zhou. 2011. Using Cross-Entity Inference to Improve Event Extraction. In *Proceedings of ACL-2011*, pp. 1127-1136.
- Ismail A., and S. Manandhar. 2010. Bilingual Lexicon Extraction from Comparable Corpora Using In-domain Terms. In *Proceedings of COLING-2010*, pp.481-489.
- Ji H. 2009. Cross-lingual Predicate Cluster Acquisition to Improve Bilingual Event Extraction by Inductive Learning. In *Proceedings of the Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics*, pp. 27-35.
- Ji H, and R. Grishman. 2008. Refining Event Extraction through Cross-Document Inference. In *Proceedings of ACL-2008*, pp. 254-262.
- Koehn P., F. Och, and D. Marcu. 2003. Statistical Phrase-based Translation. In *Proceedings of HTL-NAACL-2003*, pp. 127-133.
- Li P., and G. Zhou. 2012. Employing Morphological Structures and Sememes for Chinese Event Extraction. In *Proceedings of COLING-2012*, pp. 1619-1634.
- Li P., Q. Zhu and G. Zhou. 2013. Using Compositional Semantics and Discourse Consistency to Improve Chinese Trigger Identification. In *Proceedings of COLING-2013*, pp. 399-415.
- Li Q, H Ji, and H. Liang. 2013. Joint Event Extraction via Structured Prediction with Global Features. In *Proceedings of ACL-2013*, pp. 73-82.
- Li S, R Wang, H Liu, and CR Huang. 2013. Active Learning for Cross-Lingual Sentiment Classification. In *Proceedings of Natural Language Processing and Chinese Computing*, pp. 236-246.
- Liao S and R. Grishman. 2010. Using Document Level Cross-event Inference to Improve Event Extraction. In *Proceedings of ACL-2010*, pp. 789-797.
- Lu B., C. Tan, C. Cardie and B. K. Tsou. 2011. Joint Bilingual Sentiment Classification with Unlabeled Parallel Corpora. In *Proceedings of ACL-2011*, pp. 320-330.
- Och F., C. Tillmann, and H. Ney. 1999. Improved Alignment Models for Statistical Machine Translation. In *Proceedings of EMNLP-1999*, pp.20-28.
- Tan H., T. Zhao, and J. Zheng. 2008. Identification of Chinese Event and Their Argument Roles. In *Proceedings of CITWORKSHOPS-2008*, pp. 14-19.
- Wan X. 2008. Using Bilingual Knowledge and Ensemble Techniques for Unsupervised Chinese Sentiment Analysis. In *Proceedings of EMNLP-2008*, pp. 553-561.
- Zhao Y., Y. Wang, B. Qin, et al. 2008. Research on Chinese Event Extraction. In *Proceedings of Journal of Chinese Information*, 22(01), pp. 3-8.

Understanding Relation Temporality of Entities

Taesung Lee and Seung-won Hwang

Department of Computer Science and Engineering
Pohang University of Science and Technology (POSTECH)
Pohang, Republic of Korea
{elca4u, swhwang}@postech.edu

Abstract

This paper demonstrates the importance of relation equivalence for entity translation pair discovery. Existing approach of understanding relation equivalence has focused on using explicit features of co-occurring entities. In this paper, we explore latent features of temporality for understanding relation equivalence, and empirically show that the explicit and latent features complement each other. Our proposed hybrid approach of using both explicit and latent features improves relation translation by 0.16 F1-score, and in turn improves entity translation by 0.02.

1 Introduction

Understanding relations is important in entity tasks. In this paper, we illustrate such importance using named entity (NE) translation mining problem. Early research on NE translation used phonetic similarities, for example, to mine the translation ‘Mandelson’ → ‘曼德尔森’ [ManDeErSen] with similar sounds (Knight and Graehl, 1998; Wan and Verspoor, 1998). However, not all NE translations are based on transliterations, but they might be based on semantics (e.g., ‘WTO’ → ‘世贸组织’ [ShiMaoZuZhi]), or even arbitrary (e.g., ‘Jackie Chan’ → ‘成龙’ [ChengLong]).

To address this challenge, current state-of-the-art approaches build an entity graph for each language corpus, and align the two graphs by propagating the seed translation similarities (Figure 1) (Kim et al., 2011; You et al., 2012). For example, arbitrary translation pair such as (Jackie Chan, 成龙) can be obtained, if he is connected to his film ‘Drunken Master’ (醉拳) in both graphs. That is, we can propagate the seed translation similarity of (Drunken Master, 醉拳) to neighbor entities ‘Jackie Chan’ and ‘成龙’ in each graph.

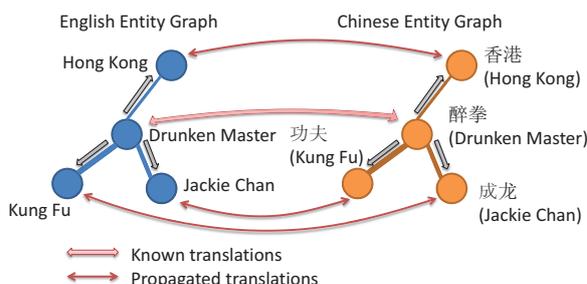


Figure 1: Entity translation by propagation.

When two graphs are obtained from parallel corpora, graphs are symmetric and “blind propagation” described above is effective. In contrast, Lee and Hwang (2013) propose “selective propagation” for asymmetric graphs, of comparing the semantics of relations. A key contribution of this paper is using relation temporality for determining relation equivalence. Existing work (Nakashole et al., 2012; Mohamed et al., 2011; Lee and Hwang, 2013) uses only co-occurring entity pairs, or explicit features (EF). For example, for a relation *pay an official visit to*, with a statement (Bush, pay an official visit to, China), an entity pair (Bush, China) is in the “support set”, which is a set of co-occurring entity pairs of *pay an official visit to*. When its support set is {(Bush, China), (Mandelson, Moscow), (Rice, Israel)}, and that of *visit* is {(Bush, China), (Rice, Israel), (Medvedev, Cuba)}, we can infer their semantic equivalence based on the set intersection: {(Bush, China), (Rice, Israel)}.

In contrast, we propose to explore corpus latent features (LF), to complement the sparsity problem of EF: Out of 158 randomly chosen correct relation translation pairs we labeled, 64% has only one co-occurring entity pair, which makes EF not very effective to identify these relation translations. Therefore, we leverage *relation temporality*, which is both orthogonal and complementary to existing efforts leveraging *entity temporality* (Kle-

mentiev and Roth, 2006; Kim et al., 2012; You et al., 2013). In particular, we discover three new challenges on using temporality for relation understanding in comparable corpora, which we discuss in detail in Section 3.2. Based on these challenges, we identify three new features for LF.

We observe the complementary nature of EF and LF, then propose a hybrid approach combining both features. Our new hybrid approach significantly improves the relation translation (0.16 higher F1-score than EF), and in turn improves the entity translation (0.02 higher F1-score).

2 Preliminary: Entity Translation by Selective Propagation

Selective propagation, leveraging the statements extracted from bilingual comparable corpora, can be summarized by several steps.

STEP 1 Initialize entity translation function $T_N^{(0)}$.

STEP 2 Build relation translation function $T_R^{(t)}$ using $T_N^{(t)}$.

STEP 3 Update entity translation function to acquire $T_N^{(t+1)}$ using $T_R^{(t)}$.

STEP 4 Repeat **STEP 2** and **STEP 3**.

For **STEP 1**, an existing method for entity translation is adopted. In our experiments, we use a non-selective (hence not requiring relation translations) propagation approach (You et al., 2012) with (Lam et al., 2007) for a base translation matrix. The focus of this paper is **STEP 2**, building the translation score $T_R^{(t)}(r_E, r_C)$ of English relation r_E and Chinese relation r_C : We will discuss the detailed procedure of **STEP 2** and propose how to improve it in Section 3. **STEP 3** is the stage that selective propagation takes place.

STEP 2 and **STEP 3** reinforce each other to improve the final entity translation function. While **STEP 3** is well-defined in (Lee and Hwang, 2013), to propagate entity translation scores when the relation semantics of the edges are equivalent, **STEP 2** has been restricted to the explicit feature, i.e., co-occurring entities or shared context. In clear contrast, by discovering novel latent features based on temporal properties, we can increase the accuracy of both entity and relation translations. Note that we omit t for readability in the following sections.

3 Relation Translation

In this section, we present our approaches to obtain relations of equivalent semantics across languages (e.g., *visit* → 访问). Formally, our goal is to build the relation translation score function $T_R(r_E, r_C)$ for English relation r_E and Chinese relation r_C .

3.1 Baseline: Explicit Feature Approach (EF)

In this section, we briefly illustrate a baseline method EF (Lee and Hwang, 2013). As we mentioned in the introduction, traditional approaches leverage common co-occurring entity-pairs. This observation also holds in the bilingual environment by exploiting seed entity translations. For example, let us say that we have two extracted statements: (Bruce Willis, star in, The Sixth Sense) and (布鲁斯·威利斯 (Bruce Willis), 主演 (star in), 第六感 (The Sixth Sense)). Knowing a few seed entity translations using T_N , ‘Bruce Willis’ → ‘布鲁斯·威利斯’ and ‘The Sixth Sense’ → ‘第六感’, we can find *star in* and *主演* are semantically similar.

Specifically, we quantify this similarity based on the number of such common entity pairs that we denote as $|H(r_E, r_C)|$ for an English relation r_E and a Chinese relation r_C . The existing approaches are variations of using $|H(r_E, r_C)|$. Our baseline implementation uses the one by (Lee and Hwang, 2013), and we refer the reader to the paper for formal definitions and processing steps we omitted due to the page limit.

Unfortunately, this approach suffers from sparsity of the common entity pairs due to the incomparability of the corpora and those entities that cannot be translated by T_N . Therefore, we leverage corpus latent features as an additional signal to overcome this problem.

3.2 Latent Feature Approach (LF)

Temporal Feature Discovery

We exploit the temporal distribution $d[x](t)$ of textual element x during t -th week in statements; we count the occurrences of the element x on a weekly basis, and normalize them to observe $\sum_t d[x](t) = 1$. For example, Figure 2a shows the relation temporal distribution $d[\text{visit}](t)$ against week t . Unlike entities, we can easily observe the dissimilarity of the temporal distributions of semantically equivalent relations. We identify the

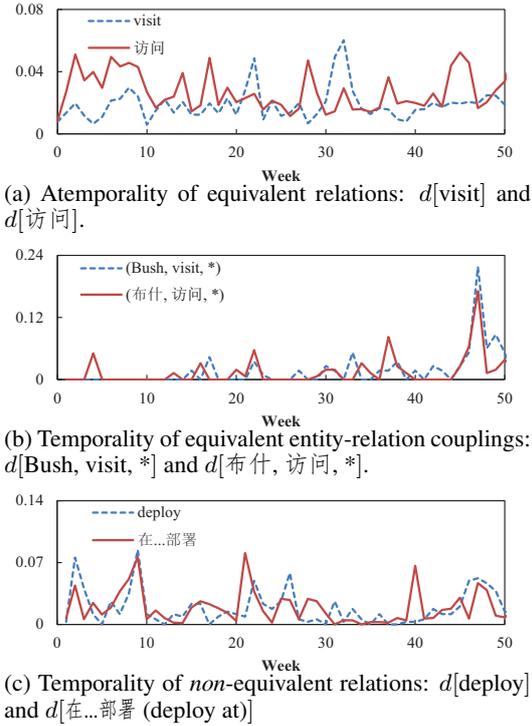


Figure 2: Temporal distributions of a relation, and a coupling.

three big challenges in exploiting the temporality in relation translations.

[C1] Considering temporal distributions $d[r]$ of relations alone is not sufficient. For relations, such as visit, that involves diverse entities, the temporal distributions are highly noisy (Figure 2a).

To address the first challenge, we use a finer-granularity unit for observing the temporality. More specifically, we exploit a coupling of a relation and an entity: $d[e, r, *]$ where e is an entity, r a relation, and $*$ is a placeholder indicating that any noun phrase is accepted for the second argument of a statement.¹ As shown in Figure 2b, $d[e, r, *]$ is more distinctive and hence a key clue to find semantically equivalent relations.

[C2] Considering entity-relation coupling distribution $d[e, r, *]$ alone is not sufficient due to the domination of individual temporality. For example, Figure 3 shows *entity-dominating* entity-relation temporality. If an entity has a peak at some period (Figure 3a), most relations that are coupled with the entity would have a peak at the very same period (Figure 3b). This makes all relations that appear with this entity very similar to

¹We use both $d[e, r, *]$ and $d[* , r, e]$ to measure the relation translation scores and leverage the average score. But in this section, we only use $d[e, r, *]$ for readability.

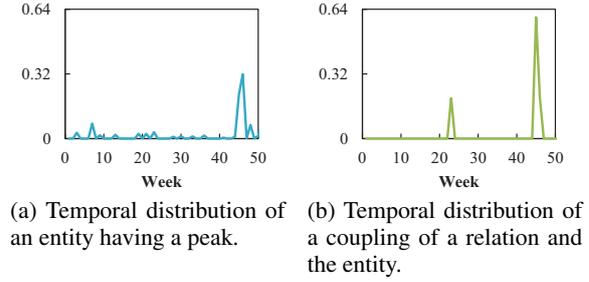


Figure 3: False positive peak of an entity-relation coupling.

each other regardless of semantics. To address this challenge, we use features to measure whether $d[e, r, *]$ is too close to either of $d[e]$ or $d[r]$.

[C3] Lastly, we have to eliminate false positives in relation temporality. To illustrate, two relations deploy and 在...部署 (deploy at) have similar temporal behaviors (Figure 2c). However, the first relation takes [person], but the second relation [location] for the second argument.

To address this, we check the common co-occurring entity pair of the relations. For example, we can obtain “Russia deployed an aircraft carrier”, but not “Russia deployed at (在...部署) an aircraft carrier”. Thus, we cannot acquire any common entity pair like (Russia, aircraft carrier) for deploy and 在...部署 (deploy at).

Relation Similarity Computation

We compute the similarity of two relations r_E in English and r_C in Chinese using the following 2-steps.

- Compute the similarity $S_{CP}(r_E, r_C, e_E, e_C)$ of temporal distributions of entity-relation couplings for each bilingual entity pair (e_E, e_C) .
- Compute the translation score $T_R^{LF}(r_E, r_C)$ by aggregating the coupling similarities.

Considering the three challenges, we produce a list of features $\{f_x(r_E, r_C, e_E, e_C)\}$ to measure the coupling similarity $S_{CP}(r_E, r_C, e_E, e_C)$ as follows.

- [Base feature] f_{ET} : $T_N(e_E, e_C)$. The entity translation score obtained in the previous iteration or the seed entity translation score.
- [C1] f_{ER} : $1 - JSD(d[e_E, r_E, *], d[e_C, r_C, *])$. The temporal similarity of the couplings, where $JSD(P, Q)$ is the Jensen-Shannon divergence of two distributions P and Q , defined as $JSD(P, Q) = \frac{1}{2}D(P||M) + \frac{1}{2}D(Q||M)$,

with $M = \frac{1}{2}(P + Q)$ and $D(P||M) = \sum_i P(i) \log \frac{P(i)}{M(i)}$.

- [C2] $f_{D1,E}, f_{D2,E}, f_{D1,C}, f_{D2,C}$:

$$JSD(d[e_E], d[e_E, r_E, *]), JSD(d[r_E], d[e_E, r_E, *])$$

$$JSD(d[e_C], d[e_C, r_C, *]), JSD(d[r_C], d[e_C, r_C, *])$$

Entity to entity-relation distribution difference (D1) and relation to entity-relation distribution difference (D2), for English and Chinese respectively.

- [C3] f_{EX} : The existence of a common entity pair using the seed entity translations (boolean). That is, $f_{EX} = 1$ if $|H(r_E, r_C)| \geq 1$, and $f_{EX} = 0$ otherwise.

Additionally, we use the following features to consider absolute frequencies $freq(\cdot)$ of textual elements as well because 1) we are more confident with more evidence and 2) in the comparable corpora, the equivalent elements are likely to show similar frequencies.

- $f_{FW,E}, f_{FW,C}$: $\mathcal{S}(freq(e_E, r_E))$ and $\mathcal{S}(freq(e_C, r_C))$. $\mathcal{S}(x)$ is a normalization function, for which we use a sigmoid function over a linear transformation of x .
- f_{FS1} and f_{FS2} :

$$\frac{\min(freq(e_E, r_E), freq(e_C, r_C))}{\max(freq(e_E, r_E), freq(e_C, r_C))},$$

$$\frac{\min(freq(r_E), freq(r_C))}{\max(freq(r_E), freq(r_C))}$$

With these features, we measure the similarity of a pair of couplings as follows.

$$S_{CP}(r_E, r_C, e_E, e_C) = \prod_x f_x(r_E, r_C, e_E, e_C) \quad (1)$$

By aggregating coupling similarities, we measure the translation score of two relations:

$$T_R^{LF}(r_E, r_C) = \sum_{(e_E, e_C) \in T} S_{CP}(r_E, r_C, e_E, e_C) \quad (2)$$

where $T = \{(e_E, e_C) | T_N(e_E, e_C) \geq \theta\}$ with $\theta = 0.6$, a set of translation pairs obtained in the seeds or previous iteration such as (Bush, 布什).

We normalize the obtained function values for each English relations using the top- k Chinese translations. That is, for (r_E, r_C) , we redefine the score as $T_R^{LF}(r_E, r_C) / \sum_{i \in [1, k]} T_R^{LF}(r_E, r_C^{rank_i})$ where $r_C^{rank_i}$ is the i -th rank Chinese relation for r_E by Equation 2. We empirically set $k = 4$.

English	LF	EF
visit	访问 (visit)	访问 (visit)
support	向...提供 (provide to ...)	-
ratify	讨论 (discuss) ²	批准 (ratify)

Table 1: Examples of relation translations.

Method	Person			Organization		
	P.	R.	F1	P.	R.	F1
LF+EF	0.84	0.80	0.82	0.60	0.52	0.56
EF	0.81	0.79	0.80	0.56	0.52	0.54
Seed	0.80	0.77	0.78	0.49	0.44	0.46
PH+SM	0.59	0.59	0.59	0.29	0.29	0.29

Table 2: Entity translation comparison.

3.3 Hybrid Approach LF+EF

We find that LF and EF are complementary. Table 1 shows the examples of relations and their translations. In general, LF can translate more relations (e.g., support and capture). However, in some cases like ratify, highly *related* relations may induce noise. That is, we always 讨论 (discuss) before we 批准 (ratify) something and hence the temporal behavior of 讨论 (discuss) is also very similar to that of ratify. On the other hand, it can be correctly translated using EF.

Thus, we produce the hybrid relation translation, and we empirically set $\lambda = 0.4$:

$$T_R^{LF+EF}(r_E, r_C) = \lambda T_R^{LF}(r_E, r_C) + (1 - \lambda) T_R^{EF}(r_E, r_C) \quad (3)$$

4 Evaluation

In this section, we evaluate the proposed approach on the entity translation task and the relation translation task. We extract English and Chinese statements from news articles in 2008 by Xinhua news who publishes news in both English and Chinese, which were also used by Lee and Hwang (2013). The number of English articles is 100,746, and that of Chinese articles is 88,031. As we can see from the difference in the numbers of the documents, the news corpora are not direct translations, but they have asymmetry of entities and relations.

4.1 Entity Translation

In this section, we present experimental settings and results on translating entities using our proposed approaches. To measure the effectiveness,

²The correct translation 批准 (ratify) is ranked second.

Methods	Precision	Recall	F1
LF+EF	0.37	0.44	0.40
LF	0.26	0.25	0.26
EF	0.41	0.17	0.24

Table 3: Relation translation comparison.

we use a set of gold standard entity translation pairs which consist of 221 person entities and 52 organization entities. We measure precision, recall, and F1-score based on the returned translation pairs for each English entity as it is done in (Lee and Hwang, 2013).

We compare our hybrid approach, which is denoted by LF+EF with EF (Lee and Hwang, 2013), a combined approach PH+SM of phonetic similarity and letter-wise semantic translation for better accuracy for organizations (Lam et al., 2007), and the seed translations Seed that we adopt (You et al., 2012) with PH+SM as a base translation matrix.³ We process one iteration of the entire framework (STEP 1-3) for both LF+EF and EF.

Table 2 shows the comparison of the methods. Our proposed approach LF+EF shows higher F1-score than the baselines. In particular, our approach outperforms EF. For example, ‘Matthew Emmons’ is a lesser known entity, and we have only few statements mentioning him in the corpora. The corpus explicit feature EF alone cannot translate the relation win and, in turn, ‘Matthew Emmons’. However, LF+EF translates him correctly into 马修·埃蒙斯 through the relation win.

4.2 Relation Translation

This section considers the relation translation task. Each relation translation method translates an English relation r_E into a list of Chinese relations, and we check whether a Chinese relation r_C with the highest translation score is the correct translation. We consider the relation translation is correct when the semantics are equivalent. For example, 去 (leave for/go to) is a correct translation of leave for, but 离开 (leave) is *not*. Total 3342 English-Chinese relation translation pairs returned by our method and the baselines are randomly shown and labeled. Out of 3342 pairs, 399 are labeled as correct.

³Our results leveraging relational temporality outperforms the reported results using entity temporality on the same data set. The two approaches using temporality are orthogonal and can be aggregated, which we leave as our future directions.

Eng. Rel.	C1	C1+C2	C1+C2+C3	EF
visit	15	4	1	1
drop	21	14	1	-
capture	6	4	1	-

Table 4: Rank of correct relation translation. The symbol ‘-’ indicates no correct translation.

Table 3 shows the comparisons of LF, EF and their hybrid LF+EF. We can clearly see that LF shows higher recall than EF while EF shows higher precision. As we emphasized in Section 3.3, we can see their complementary property. Their hybrid LF+EF has both high precision and recall, thus has the highest F1-score.

Note that the absolute numbers (due to the harsh evaluation criteria) may look low. But the top translations are still relevant (e.g., fight is translated to 驻 (deploy troops)). In addition, the lower ranked but correct relation translations also affect entity translation. Therefore, even lower-performing EF boosted the entity translations, and in effect, our approach could achieve higher F1-score in the entity translation task.

To illustrate the detailed effects of the corpus latent features, Table 4 shows the ranks of correct Chinese translations for English relations by methods using selected features for the challenges. For comparison, the ranks of the correct translations when using EF are shown. Our approach using the entity-relation coupling similarity feature for [C1] alone often cannot find the correct translations. But using all features removes such noise.

5 Conclusion

This paper studied temporality features for relation equivalence. With the proposed features, we devised a hybrid approach combining corpus latent and explicit features with complementary strength. We empirically showed the effectiveness of our hybrid approach on relation translation, and it, in turn, improved entity translation.

Acknowledgments

This research was supported by the MSIP (The Ministry of Science, ICT and Future Planning), Korea and Microsoft Research, under IT/SW Creative research program supervised by the NIPA(National IT Industry Promotion Agency). (NIPA-2013-H0503-13-1009).

References

- Jinhan Kim, Long Jiang, Seung-won Hwang, Young-In Song, and Ming Zhou. 2011. Mining entity translations from comparable corpora: a holistic graph mapping approach. In *Proc. 20th ACM International Conference on Information and Knowledge Management (CIKM 2011)*, pages 1295–1304. ACM.
- Jinhan Kim, Seung-won Hwang, Long Jiang, Young-In Song, and Ming Zhou. 2012. Entity translation mining from comparable corpora: Combining graph mapping with corpus latent features. In *IEEE Transactions on Knowledge and Data Engineering*, pages 1787–1800.
- Alexandre Klementiev and Dan Roth. 2006. Named entity transliteration and discovery from multilingual comparable corpora. In *Proc. 8th Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2006)*, pages 82–88. Association for Computational Linguistics.
- Kevin Knight and Jonathan Graehl. 1998. Machine transliteration. *Computational Linguistics*, 24(4):599–612, December.
- Wai Lam, Shing-Kit Chan, and Ruizhang Huang. 2007. Named entity translation matching and learning: With application for mining unseen translations. *ACM Transactions on Information Systems*, 25(1), February.
- Taesung Lee and Seung-won Hwang. 2013. Bootstrapping entity translation on weakly comparable corpora. In *Proc. 51st Annual Meeting on Association for Computational Linguistics (ACL 2013)*. Association for Computational Linguistics.
- Thahir Mohamed, Estevam Hruschka, and Tom Mitchell. 2011. Discovering relations between noun categories. In *Proc. 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 1447–1455. Association for Computational Linguistics.
- Ndapandula Nakashole, Gerhard Weikum, and Fabian M. Suchanek. 2012. PATTY: A Taxonomy of Relational Patterns with Semantic Types. In *Proc. 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*. Association for Computational Linguistics.
- Stephen Wan and Cornelia Maria Verspoor. 1998. Automatic English-Chinese name transliteration for development of multilingual resources. In *Proc. 36th Annual Meeting on Association for Computational Linguistics (ACL 1998) and 17th International Conference on Computational Linguistics (COLING 1998)*, pages 1352–1356. Association for Computational Linguistics.
- Gae-won You, Seung-won Hwang, Young-In Song, Long Jiang, and Zaiqing Nie. 2012. Efficient entity translation mining: A parallelized graph alignment approach. *ACM Transactions on Information Systems*, 30(4):25:1–25:23, November.
- Gae-won You, Young-rok Cha, Jinhan Kim, and Seung-won Hwang. 2013. Enriching entity translation discovery using selective temporality. In *Proc. 51st Annual Meeting on Association for Computational Linguistics (ACL 2013)*, pages 201–205. Association for Computational Linguistics.

Does the Phonology of L1 Show Up in L2 Texts?

Garrett Nicolai and Grzegorz Kondrak

Department of Computing Science

University of Alberta

{nicolai, gkondrak}@ualberta.ca

Abstract

The relative frequencies of character bigrams appear to contain much information for predicting the first language (L1) of the writer of a text in another language (L2). Tsur and Rappoport (2007) interpret this fact as evidence that word choice is dictated by the phonology of L1. In order to test their hypothesis, we design an algorithm to identify the most discriminative words and the corresponding character bigrams, and perform two experiments to quantify their impact on the L1 identification task. The results strongly suggest an alternative explanation of the effectiveness of character bigrams in identifying the native language of a writer.

1 Introduction

The task of Native Language Identification (NLI) is to determine the first language of the writer of a text in another language. In a ground-breaking paper, Koppel et al. (2005) propose a set of features for this task: function words, character n -grams, rare part-of-speech bigrams, and various types of errors. They report 80% accuracy in classifying a set of English texts into five L1 languages using a multi-class linear SVM.

The First Shared Task on Native Language Identification (Tetreault et al., 2013) attracted submissions from 29 teams. The accuracy on a set of English texts representing eleven L1 languages ranged from 31% to 83%. Many types of features were employed, including word length, sentence length, paragraph length, document length, sentence complexity, punctuation and capitalization, cognates, dependency parses, topic models, word suffixes, collocations, function word n -grams, skip-grams, word networks, Tree Substitution Grammars, string kernels, cohesion, and

passive constructions (Abu-Jbara et al., 2013; Li, 2013; Brooke and Hirst, 2013; Cimino et al., 2013; Daudaravicius, 2013; Goutte et al., 2013; Henderson et al., 2013; Hladka et al., 2013; Bykh et al., 2013; Lahiri and Mihalcea, 2013; Lynam, 2013; Malmasi et al., 2013; Mizumoto et al., 2013; Nicolai et al., 2013; Popescu and Ionescu, 2013; Swanson, 2013; Tsvetkov et al., 2013). In particular, word n -gram features appear to be particularly effective, as they were used by the most competitive teams, including the one that achieved the highest overall accuracy (Jarvis et al., 2013). Furthermore, the most discriminative word n -grams often contained the name of the native language, or countries where it is commonly spoken (Gebre et al., 2013; Malmasi et al., 2013; Nicolai et al., 2013). We refer to such words as *toponymic terms*.

There is no doubt that the toponymic terms are useful for increasing the NLI accuracy; however, from the psycho-linguistic perspective, we are more interested in what characteristics of L1 show up in L2 texts. Clearly, L1 affects the L2 writing in general, and the choice of words in particular, but what is the role played by the phonology? Tsur and Rappoport (2007) observe that limiting the set of features to the relative frequency of the 200 most frequent character bigrams yields a respectable 66% accuracy on a 5-language classification task. The authors propose the following hypothesis to explain this finding: “*the choice of words [emphasis added] people make when writing in a second language is strongly influenced by the phonology of their native language*”. As the orthography of alphabetic languages is at least partially representative of the underlying phonology, character bigrams may capture these phonological preferences.

In this paper, we provide evidence against the above hypothesis. We design an algorithm to identify the most discriminative words and the character bigrams that are indicative of such words,

and perform two experiments to quantify their impact on the NLI task. The results of the first experiment demonstrate that the removal of a relatively small set of discriminative words from the training data significantly impairs the accuracy of a bigram-based classifier. The results of the second experiment reveal that the most indicative bigrams are quite similar across different language sets. We conclude that character bigrams are effective in determining L1 of the author because they reflect differences in L2 word usage that are unrelated to the phonology of L1.

2 Method

Tsur and Rappoport (2007) report that character bigrams are more effective for the NLI task than either unigrams or trigrams. We are interested in identifying the character bigrams that are indicative of the most discriminative words in order to quantify their impact on the bigram-based classifier.

We follow both Koppel et al. (2005) and Tsur and Rappoport (2007) in using a multi-class SVM classifier for the NLI task. The classifier computes a weight for each feature coupled with each L1 language by attempting to maximize the overall accuracy on the training set. For example, if we train the classifier using words as features, with values representing their frequency relative to the length of the document, the features corresponding to the word *China* might receive the following weights:

Arabic	Chinese	Hindi	Japanese	Telugu
-770	1720	-276	-254	-180

These weights indicate that the word provides strong positive evidence for Chinese as L1, as opposed to the other four languages.

We propose to quantify the importance of each word by converting its SVM feature weights into a single score using the following formula:

$$WordScore_i = \sqrt{\sum_{j=1}^N w_{ij}^2}$$

where N is the number of languages, and w_{ij} is the feature weight of word i in language j . The formula assigns higher scores to words with weights of high magnitude, either positive or negative. We use the Euclidean norm rather than the

Algorithm 1 Computing the scores of words and bigrams in the data.

```

1: create list of words in training data
2: train SVM using words as features
3: for all words  $i$  do
4:    $WordScore_i = \sqrt{\sum_{j=1}^N w_{ij}^2}$ 
5: end for
6: sort words by WordScore
7: NormValue = WordScore200
8: create list of 200 most frequent bigrams
9: for bigrams  $k = 1$  to 200 do
10:   $BigramScore_k = \prod_{k \in i} \frac{WordScore_i}{NormValue}$ 
11: end for
12: sort character bigrams by BigramScore

```

sum of raw weights because we are interested in the discriminative power of the words.

We normalize the word scores by dividing them by the score of the 200th word. Consequently, only the top 200 words have scores greater than or equal to 1.0. For our previous example, the 200th word has a word score of 1493, while *China* has a word score of 1930, which is normalized to $1930/1493 = 1.29$. On the other hand, the 1000th word gets a normalized score of 0.43.

In order to identify the bigrams that are indicative of the most discriminative words, we promote those that appear in the high-scoring words, and downgrade those that appear in the low-scoring words. Some bigrams that appear often in the high-scoring words may be very common. For example, the bigram *an* occurs in words like *Japan*, *German*, and *Italian*, but also by itself as a determiner, as an adjectival suffix, and as part of the conjunction *and*. Therefore, we calculate the importance score for each character bigram by multiplying the scores of each word in which the bigram occurs.

Algorithm 1 summarizes our method of identifying the discriminative words and indicative character bigrams. In line 2, we train an SVM on the words encountered in the training data. In lines 3 and 4, we assign the Euclidean norm of the weight vector of each word as its score. Starting in line 7, we determine which character bigrams are representative of high scoring words. In line 10, we calculate the bigram scores.

3 Experiments

In this section, we describe two experiments aimed at quantifying the importance of the discriminative words and the indicative character bigrams that are identified by Algorithm 1.

3.1 Data

We use two different NLI corpora. We follow the setup of Tsur and Rappoport (2007) by extracting two sets, denoted I1 and I2 (Table 1), from the International Corpus of Learner English (ICLE), Version 2 (Granger et al., 2009). Each set consists of 238 documents per language, randomly selected from the ICLE corpus. Each of the documents corresponds to a different author, and contains between 500 and 1000 words. We follow the methodology of the paper in performing 10-fold cross-validation on the sets of languages used by the authors.

For the development of the method described in Section 2, we used a different corpus, namely the TOEFL Non-Native English Corpus (Blanchard et al., 2013). It consists of essays written by native speakers of eleven languages, divided into three English proficiency levels. In order to maintain consistency with the ICLE sets, we extracted three sets of five languages apiece (Table 1), with each set including both related and unrelated languages: European languages that use Latin script (T1), non-European languages that use non-Latin scripts (T2), and a mixture of both types (T3). Each sub-corpus was divided into a training set of 80%, and development and test sets of 10% each. The training sets are composed of approximately 700 documents per language, with an average length of 350 words per document. There are over 5000 word types per language, and over 1000 character bigrams in total. The test sets include approximately 90 documents per language. We report results on the test sets, after training on both the training and development sets.

3.2 Setup

We replicate the experiments of Tsur and Rappoport (2007) by limiting the features to the 200 most frequent character bigrams.¹ The feature values are set to the frequency of the character bi-

¹Our development experiments suggest that using the full set of bigrams results in a higher accuracy of a bigram-based classifier. However, we limit the set of features to the 200 most frequent bigrams for the sake of consistency with previous work.

ICLE:	
I1	Bulgarian Czech French Russian Spanish
I2	Czech Dutch Italian Russian Spanish
TOEFL:	
T1	French German Italian Spanish Turkish
T2	Arabic Chinese Hindi Japanese Telugu
T3	French German Japanese Korean Telugu

Table 1: The L1 language sets.

grams normalized by the length of the document. We use these feature vectors as input to the SVM-Multiclass classifier (Joachims, 1999). The results are shown in the *Baseline* column of Table 2.

3.3 Discriminative Words

The objective of the first experiment is to quantify the influence of the most discriminative words on the accuracy of the bigram-based classifier. Using Algorithm 1, we identify the 100 most discriminative words, and remove them from the training data. The bigram counts are then recalculated, and the new 200 most frequent bigrams are used as features for the character-level SVM. Note that the number of the features in the classifier remains unchanged.

The results are shown in the *Discriminative Words* column of Table 2. We see a statistically significant drop in the accuracy of the classifier with respect to the baseline in all sets except T3. The words that are identified as the most discriminative include function words, punctuation, very common content words, and the toponymic terms. The 10 highest scoring words from T1 are: *indeed*, *often*, *statement*, *:* (colon), *question*, *instance*, *...* (ellipsis), *opinion*, *conclude*, and *however*. In addition, *France*, *Turkey*, *Italian*, *Germany*, and *Italy* are all found among the top 70 words.

For comparison, we attempt to quantify the effect of removing the same number of randomly-selected words from the training data. Specifically, we discard all tokens that correspond to 100 word types that have the same or slightly higher frequency as the discriminative words. The results are shown in the *Random Words* column of Table 2. The decrease is much smaller for I1, I2, and T1, while the accuracy actually increases for T2 and T3. This illustrates the impact that the most discriminative words have on the bigram-based classifier beyond simple reduction in the amount of the training data.

Set	Baseline	Random Words	Discriminative Words	Random Bigrams	Indicative Bigrams
I1	67.5	-0.2	-3.6	-1.0	-2.2
I2	66.9	-2.5	-5.5	-0.7	-2.8
T1	60.7	-3.3	-7.7	-2.5	-3.9
T2	60.6	+0.5	-3.8	-1.1	-5.9
T3	62.2	+0.3	-0.0	-0.5	-4.1

Table 2: The impact of subsets of word types and bigram features on the accuracy of a bigram-based NLI classifier.

3.4 Indicative Bigrams

Using Algorithm 1, we identify the top 20 character bigrams, and replace them with randomly selected bigrams. The results of this experiment are reported in the *Indicative Bigrams* column of Table 2. It is to be expected that the replacement of any 20 of the top bigrams with 20 less useful bigrams will result in some drop in accuracy, regardless of which bigrams are chosen for replacement. For comparison, the *Random Bigrams* column of Table 2 shows the mean accuracy over 100 trials obtained when 20 bigrams randomly selected from the set of 200 bigrams are replaced with random bigrams from outside of the set.

The results indicate that our algorithm indeed identifies 20 bigrams that are on average more important than the other 180 bigrams. What is really striking is that the sets of 20 indicative character bigrams overlap substantially across different sets. Table 3 shows 17 bigrams that are common across the three TOEFL corpora, ordered by their score, together with some of the highly scored words in which they occur. Four of the bigrams consist of punctuation marks and a space.² The remaining bigrams indicate function words, toponymic terms like *Germany*, and frequent content words like *take* and *new*.

The situation is similar in the ICLE sets, where likewise 17 out of 20 bigrams are common. The inter-fold overlap is even greater, with 19 out of 20 bigrams appearing in each of the 10 folds. In particular, the bigrams *fr* and *bu* can be traced to both the function words *from* and *but*, and the presence of French and Bulgarian in I1. However, the fact that the two bigrams are also on the list for

²It appears that only the relatively low frequency of most of the punctuation bigrams prevents them from dominating the sets of the indicative bigrams. When using all bigrams instead of the top 200, the majority of the indicative bigrams contain punctuation.

Bigram	Words
→	
'	
..	
..	
u_	you Telugu
f_	of
ny	any many Germany
yo	you your
w_	now how
i_	I
-y	you your
ew	new knew
kn	know knew
ey	they Turkey
wh	what why where <i>etc.</i>
of	of
ak	make take

Table 3: The most indicative character bigrams in the TOEFL corpus (sorted by score).

the I2 set, which does not include these languages, suggests that their importance is mostly due to the function words.

3.5 Discussion

In the first experiment, we showed that the removal of the 100 most discriminative words from the training data results in a significant drop in the accuracy of the classifier *that is based exclusively on character bigrams*. If the hypothesis of Tsur and Rappoport (2007) was true, this should not be the case, as the phonology of L1 would influence the choice of words across the lexicon.

In the second experiment, we found that the majority of the most indicative character bigrams *are shared among different language sets*. The bigrams appear to reflect primarily high-frequency function words. If the hypothesis was true, this

should not be the case, as the diverse L1 phonologies would induce different sets of bigrams. In fact, the highest scoring bigrams reflect punctuation patterns, which have little to do with word choice.

4 Conclusion

We have provided experimental evidence against the hypothesis that the phonology of L1 strongly affects the choice of words in L2. We showed that a small set of high-frequency function words have disproportionate influence on the accuracy of a bigram-based NLI classifier, and that the majority of the indicative bigrams appear to be independent of L1. This suggests an alternative explanation of the effectiveness of a bigram-based classifier in identifying the native language of a writer — that the character bigrams simply mirror differences in the word usage rather than the phonology of L1.

Our explanation concurs with the findings of Daland (2013) that unigram frequency differences in certain types of phonological segments between child-directed and adult-directed speech are due to a small number of word types, such as *you*, *what*, and *want*, rather than to any general phonological preferences. He argues that the relative frequency of sounds in speech is driven by the relative frequency of words. In a similar vein, Koppel et al. (2005) see the usefulness of character *n*-grams as “simply an artifact of variable usage of particular words, which in turn might be the result of different thematic preferences,” or as a reflection of the L1 orthography.

We conclude by noting that our experimental results do not imply that the phonology of L1 has absolutely no influence on L2 writing. Rather, they show that the evidence from the Native Language Identification task has so far been inconclusive in this regard.

Acknowledgments

We thank the participants and the organizers of the shared task on NLI at the BEA8 workshop for sharing their reflections on the task. We also thank an anonymous reviewer for pointing out the study of Daland (2013).

This research was supported by the Natural Sciences and Engineering Research Council of Canada and the Alberta Innovates Technology Futures.

References

- Amjad Abu-Jbara, Rahul Jha, Eric Morley, and Dragomir Radev. 2013. Experimental results on the native language identification shared task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 82–88.
- Daniel Blanchard, Joel Tetreault, Derrick Higgins, Aoife Cahill, and Martin Chodorow. 2013. TOEFL11: A Corpus of Non-Native English. Technical report, Educational Testing Service.
- Julian Brooke and Graeme Hirst. 2013. Using other learner corpora in the 2013 NLI shared task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 188–196.
- Serhiy Bykh, Sowmya Vajjala, Julia Krivanek, and Detmar Meurers. 2013. Combining shallow and linguistically motivated features in native language identification. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 197–206.
- Andrea Cimino, Felice Dell’Orletta, Giulia Venturi, and Simonetta Montemagni. 2013. Linguistic profiling based on general-purpose features and native language identification. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 207–215.
- Robert Daland. 2013. Variation in the input: a case study of manner class frequencies. *Journal of Child Language*, 40(5):1091–1122.
- Vidas Daudaravicius. 2013. VTEX system description for the NLI 2013 shared task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 89–95.
- Binyam Gebrekidan Gebre, Marcos Zampieri, Peter Wittenburg, and Tom Heskes. 2013. Improving native language identification with TF-IDF weighting. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 216–223.
- Cyril Goutte, Serge Léger, and Marine Carpuat. 2013. Feature space selection and combination for native language identification. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 96–100.
- Sylvaine Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot. 2009. INTERNATIONAL CORPUS OF LEARNER ENGLISH: VERSION 2.
- John Henderson, Guido Zarrella, Craig Pfeifer, and John D. Burger. 2013. Discriminating non-native English with 350 words. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 101–110.

- Barbora Hladka, Martin Holub, and Vincent Kriz. 2013. Feature engineering in the NLI shared task 2013: Charles University submission report. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 232–241.
- Scott Jarvis, Yves Bestgen, and Steve Pepper. 2013. Maximizing classification accuracy in native language identification. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 111–118.
- Thorsten Joachims. 1999. Making large-scale support vector machine learning practical. In *Advances in kernel methods*, pages 169–184. MIT Press.
- Moshe Koppel, Jonathan Schler, and Kfir Zigdon. 2005. Determining an author’s native language by mining a text for errors. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 624–628, Chicago, IL. ACM.
- Shibamouli Lahiri and Rada Mihalcea. 2013. Using n-gram and word network features for native language identification. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 251–259.
- Baoli Li. 2013. Recognizing English learners. native language from their writings. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 119–123.
- André Lynam. 2013. Native language identification using large scale lexical features. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 266–269.
- Shervin Malmasi, Sze-Meng Jojo Wong, and Mark Dras. 2013. NLI shared task 2013: MQ submission. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 124–133.
- Tomoya Mizumoto, Yuta Hayashibe, Keisuke Sakaguchi, Mamoru Komachi, and Yuji Matsumoto. 2013. NAIST at the NLI 2013 shared task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 134–139.
- Garrett Nicolai, Bradley Hauer, Mohammad Salameh, Lei Yao, and Grzegorz Kondrak. 2013. Cognate and misspelling features for natural language identification. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 140–145.
- Marius Popescu and Radu Tudor Ionescu. 2013. The story of the characters, the DNA and the native language. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 270–278.
- Ben Swanson. 2013. Exploring syntactic representations for native language identification. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 146–151.
- Joel Tetreault, Daniel Blanchard, and Aoife Cahill. 2013. A Report on the First Native Language Identification Shared Task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*.
- Oren Tsur and Ari Rappoport. 2007. Using Classifier Features for Studying the Effect of Native Language on the Choice of Written Second Language Words. In *Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition*, pages 9–16, Prague, Czech Republic.
- Yulia Tsvetkov, Naama Twitto, Nathan Schneider, Noam Ordan, Manaal Faruqui, Victor Chahuneau, Shuly Wintner, and Chris Dyer. 2013. Identifying the L1 of non-native writers: the CMU-Haifa system. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 279–287.

Cross-lingual Opinion Analysis via Negative Transfer Detection

Lin Gui^{1,2}, Ruifeng Xu^{1*}, Qin Lu², Jun Xu¹, Jian Xu², Bin Liu¹, Xiaolong Wang¹

¹Key Laboratory of Network Oriented Intelligent Computation, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055

²Department Of Computing, the Hong Kong Polytechnic University
guilin.nlp@gmail.com, xuruifeng@hitsz.edu.cn, csluqin@comp.polyu.edu.hk, xujun@hitsz.edu.cn, csjxu@comp.polyu.edu.hk, {bliu,wangxl}@insun.hit.edu.cn

Abstract

Transfer learning has been used in opinion analysis to make use of available language resources for other resource scarce languages. However, the cumulative class noise in transfer learning adversely affects performance when more training data is used. In this paper, we propose a novel method in transductive transfer learning to identify noises through the detection of negative transfers. Evaluation on NLP&CC 2013 cross-lingual opinion analysis dataset shows that our approach outperforms the state-of-the-art systems. More significantly, our system shows a monotonic increase trend in performance improvement when more training data are used.

1 Introduction

Mining opinions from text by identifying their positive and negative polarities is an important task and supervised learning methods have been quite successful. However, supervised methods require labeled samples for modeling and the lack of sufficient training data is the performance bottle-neck in opinion analysis especially for resource scarce languages. To solve this problem, the transfer leaning method (Arnold et al., 2007) have been used to make use of samples from a resource rich source language to a resource scarce target language, also known as cross language opinion analysis (CLOA).

In transductive transfer learning (TTL) where the source language has labeled data and the target language has only unlabeled data, an algorithm needs to select samples from the unlabeled target language as the training data and assign them with class labels using some estimated confidence. These labeled samples in the target language, referred to as the transferred samples, also have a probability of being misclassified. During

training iterations, the misclassification introduces class noise which accumulates, resulting in a so called negative transfer that affects the classification performance.

In this paper, we propose a novel method aimed at reducing class noise for TTL in CLOA. The basic idea is to utilize transferred samples with high quality to identify those negative transfers and remove them as class noise to reduce noise accumulation in future training iterations. Evaluations on NLP&CC 2013 CLOA evaluation data set show that our algorithm achieves the best result, outperforming the current state-of-the-art systems. More significantly, our system shows a monotonic increasing trend in performance when more training data are used beating the performance degradation curse of most transfer learning methods when training data reaches certain size.

The rest of the paper is organized as follows. Section 2 introduces related works in transfer learning, cross lingual opinion analysis, and class noise detection technology. Section 3 presents our algorithm. Section 4 gives performance evaluation. Section 5 concludes this paper.

2 Related works

TTL has been widely used before the formal concept and definition of TTL was given in (Arnold, 2007). Wan introduced the co-training method into cross-lingual opinion analysis (Wan, 2009; Zhou et al., 2011), and Aue et al. introduced transfer learning into cross domain analysis (Aue, 2005) which solves similar problems. In this paper, we will use the terms source language and target language to refer to all cross lingual/domain analysis.

Traditionally, transfer learning methods focus on how to estimate the confidence score of transferred samples in the target language or domain (Blitzer et al, 2006, Huang et al., 2007; Sugiyama et al., 2008, Chen et al, 2011, Lu et al., 2011). In some tasks, researchers utilize NLP tools such as alignment to reduce the bias towards that of

the source language in transfer learning (Meng et al., 2012). However, detecting misclassification in transferred samples (referred to as class noise) and reducing negative transfers are still an unresolved problem.

There are two basic methods for class noise detection in machine learning. The first is the classification based method (Brodley and Friedl, 1999; Zhu et al, 2003; Zhu 2004; Sluban et al., 2010) and the second is the graph based method (Zighed et al, 2002; Muhlenbach et al, 2004; Jiang and Zhou, 2004). Class noise detection can also be applied to semi-supervised learning because noise can accumulate in iterations too. Li employed Zighed’s cut edge weight statistic method in self-training (Li and Zhou, 2005) and co-training (Li and Zhou, 2011). Chao used Li’s method in tri-training (Chao et al, 2008). (Fukamoto et al, 2013) used the support vectors to detect class noise in semi-supervised learning.

In TTL, however, training and testing samples cannot be assumed to have the same distributions. Thus, noise detection methods used in semi-supervised learning are not directly suited in TTL. Y. Cheng has tried to use semi-supervised method (Jiang and Zhou, 2004) in transfer learning (Cheng and Li, 2009). His experiment showed that their approach would work when the source domain and the target domain share similar distributions. How to reduce negative transfers is still a problem in transfer learning.

3 Our Approach

In order to reduce negative transfers, we propose to incorporate class noise detection into TTL. The basic idea is to first select high quality labeled samples after certain iterations as indicator to detect class noise in transferred samples. We then remove noisy samples that cause negative transfers from the current accumulated training set to retain an improved set of training data for the remainder of the training phase. This negative sample reduction process can be repeated several times during transfer learning. Two questions must be answered in this approach: (1) how to measure the quality of transferred samples, and (2) how to utilize high quality labeled samples to detect class noise in training data.

3.1 Estimating Testing Error

To determine the quality of the transferred samples that are added iteratively in the learning process, we cannot use training error to estimate true error because the training data and the test-

ing data have different distributions. In this work, we employ the Probably Approximately Correct (PAC) learning theory to estimate the error boundary. According to the PAC learning theory, the least error boundary ϵ is determined by the size of the training set m and the class noise rate η , bound by the following relation:

$$\epsilon \propto \sqrt{1/m(1-\eta)^2} \quad (1)$$

In TTL, m increases linearly, yet η is multiplied in each iteration. This means the significance of m to performance is higher at the beginning of transfer learning and gradually slows down in later iterations. On the contrary, the influence of class noise increases. That is why performance improves initially and gradually falls to negative transfer when noise accumulation outperforms the learned information as shown in Fig.1. In TTL, transferred samples in both the training data and test data have the same distribution. This implies that we can apply the PAC theory to analyze the error boundary of the machine learning model using transferred data.

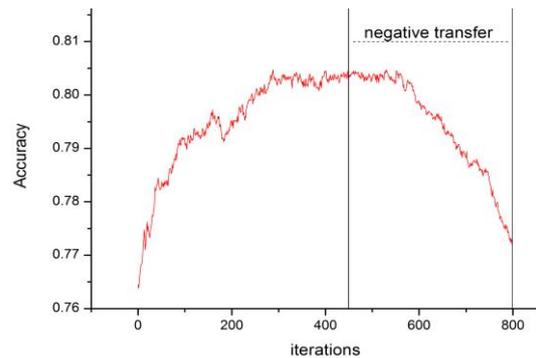


Figure 1 Negative transfer in the learning process

According to PAC theorem with an assumed fixed probability δ (Angluin and Laird, 1988), the least error boundary ϵ is given by:

$$\epsilon = \sqrt{2 \ln(2N/\delta) / (m(1-\eta)^2)} \quad (2)$$

where N is a constant decided by the hypothesis space. In any iteration during TTL, the hypothesis space is the same and the probability δ is fixed. Thus the least error boundary is determined by the size of the transferred sample m and the class noise of transferred samples η . According to (2), we apply a manifold assumption based method to estimate η . Let T be the number of iterations to serve as one period. We then estimate the least error boundary before and after each T to measure the quality of transferred samples during each T . If the least error boundary is reduced, it means that transferred samples used in this period are of high quality and can improve the performance. Otherwise, the transfer learning algorithm should stop.

3.2 Estimating Class Noise

For formula (2) to work, we need to know the class noise rate η to calculate the error boundary. Obviously, we cannot use conditional probabilities from the training data in the source language to estimate the noise rate η of the transferred samples because the distribution of source language is different from that of target language.

Consider a KNN graph on the transferred samples using any similarity metric, for example, cosine similarity, for any two connected vertex (x_i, y_i) and (x_j, y_j) in the graph from samples to classes, the edge weight is given by:

$$w_{ij} = \text{sim}(x_i, x_j) \quad (3)$$

Furthermore, a sign function for the two vertices (x_i, y_i) and (x_j, y_j) , is defined as:

$$I_{ij} = \begin{cases} 0, & \text{if } y_i = y_j \\ 1, & \text{if } y_i \neq y_j \end{cases} \quad (4)$$

According to the manifold assumption, the conditional probability $P(y_i|x_i)$ can be approximated by the frequency of $P(y_i = y_j)$ which is equal to $P(I_{ij} = 0)$. In opinion annotations, the agreement of two annotators is often no larger than 0.8. This means that for the best cases $P(I_{ij} = 1) = 0.2$. Hence I_{ij} follows a Bernoulli distribution with $p=0.2$ for the best cases in manual annotations.

Let $C_{ij} = \{(x_j, y_j)\}$ be the vertices that are connected to the i^{th} vertex, the statistical magnitude of the i^{th} vertex can be defined as:

$$J_i = \sum_j w_{ij} \cdot I_{ij} \quad (5)$$

where j refers to the j^{th} vertex that is connected to the i^{th} vertex.

From the theory of cut edge statics, we know that the expectation of J_i is:

$$\mu_i = P(I_{ij} = 1) * \sum_j w_{ij} \quad (6)$$

And the variance of J_i is:

$$\sigma_i^2 = P(I_{ij} = 0)P(I_{ij} = 1) * \sum_j w_{ij}^2 \quad (7)$$

By the Center Limit Theorem (CLT), J_i follows the normal distribution:

$$\frac{J_i - \mu_i}{\sigma_i} \sim N(0,1) \quad (8)$$

To detect the noise rate of a sample (x_i, y_i) , we can use (8) as the null hypothesis to test the significant level. Let p_i denotes probability of the correct classification for a transferred sample. p_i should follow a normal distribution,

$$p_i = \frac{1}{\sqrt{2\pi}\sigma_i} \int_{J_i}^{+\infty} e^{-\frac{(t-\mu_i)^2}{2\sigma_i^2}} dt. \quad (9)$$

Note that experiments (Li and Zhou, 2011; Cheng and Li, 2009; Brodley and Friedl, 1999) have shown that p_i is related to the error rate of

the example (x_i, y_i) , but it does not reflect the ground-truth probability in statistics. Hence we assume the class noise rate of example (x_i, y_i) is:

$$\eta_i = 1 - p_i \quad (10)$$

We take the general significant level of 0.05 to reject the null hypothesis. It means that if η_i of (x_i, y_i) is larger than 0.95, the sample will be considered as a class noisy sample. Furthermore, η_i can be used to estimate the average class noise rate of a transferred samples in (2).

In our proposed approach, we establish the quality estimate period T to conduct class noise detection to estimate the class noise rate of transferred samples. Based on the average class noise we can get the least error boundary so as to tell if an added sample is of high quality. If the newly added samples are of high quality, they can be used to detect class noise in transferred training data. Otherwise, transfer learning should stop. The flow chart for negative transfer is in Fig.2.

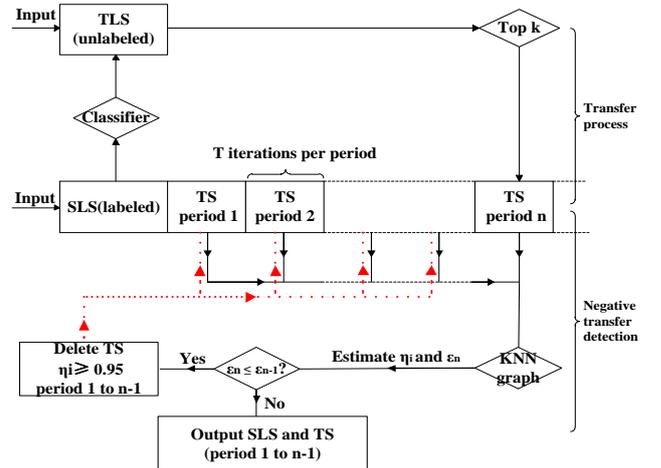


Figure 2 Flow charts of negative transfer detection

In the above flow chart, SLS and TLS refer to the source and target language samples, respectively. TS refers to the transferred samples. Let T denote quality estimate period T in terms of iteration numbers. The transfer process select k samples in each iteration. When one period of transfer process finishes, the negative transfer detection will estimate the quality by comparing and either select the new transferred samples or remove class noise accumulated up to this iteration.

4 Experiment

4.1 Experiment Setting

The proposed approach is evaluated on the NLP&CC 2013 cross-lingual opinion analysis (in

short, NLP&CC) dataset¹. In the training set, there are 12,000 labeled English Amazon.com products reviews, denoted by Train_ENG, and 120 labeled Chinese product reviews, denoted as Train_CHN, from three categories, DVD, BOOK, MUSIC. 94,651 unlabeled Chinese products reviews from corresponding categories are used as the development set, denoted as Dev_CHN. In the testing set, there are 12,000 Chinese product reviews (shown in Table.1). This dataset is designed to evaluate the CLOA algorithm which uses Train_CHN, Train_ENG and Dev_CHN to train a classifier for Test_CHN. The performance is evaluated by the correct classification accuracy for each category in Test_CHN²:

$$Accuracy_c = \frac{\#correctly\ classified\ samples\ in\ c}{4000}$$

where c is either *DVD*, *BOOK* or *MUSIC*.

Team	DVD	Book	Music
Train_CHN	40	40	40
Train_ENG	4000	4000	4000
Dev_CHN	17814	47071	29677
Test_CHN	4000	4000	4000

Table.1 The NLP&CC 2013 CLOA dataset

In the experiment, the basic transfer learning algorithm is co-training. The Chinese word segmentation tool is ICTCLAS (Zhang et al, 2003) and Google Translator³ is the MT for the source language. The monolingual opinion classifier is SVM^{light4}, word unigram/bigram features are employed.

4.2 CLOA Experiment Results

Firstly, we evaluate the baseline systems which use the same monolingual opinion classifier with three training dataset including Train_CHN, translated Train_ENG and their union, respectively.

	DVD	Book	Music	Accuracy
Train_CHN	0.552	0.513	0.500	0.522
Train_ENG	0.729	0.733	0.722	0.728
Train_CHN +Train_ENG	0.737	0.722	0.742	0.734

Table.2 Baseline performances

It can be seen that using the same method, the classifier trained by Train_CHN are on average 20% worse than the English counter parts. The combined use of Train_CHN and translated Train_ENG, however, obtained similar

performance to the English counter parts. This means the predominant training comes from the English training data.

In the second set of experiment, we compare our proposed approach to the official results in NLP&CC 2013 CLOA evaluation and the result is given in Table 3. Note that in Table 3, the top performer of NLP&CC 2013 CLOA evaluation is the HLT-HITSZ system(underscored in the table), which used the co-training method in transfer learning (Gui et al, 2013), proving that co-training is quite effective for cross-lingual analysis. With the additional negative transfer detection, our proposed approach achieves the best performance on this dataset outperformed the top system (by HLT-HITSZ) by a 2.97% which translate to 13.1% error reduction improvement to this state-of-the-art system as shown in the last row of Table 3.

Team	DVD	Book	Music	Accuracy
BUAA	0.481	0.498	0.503	0.494
BISTU	0.647	0.598	0.661	0.635
<u>HLT-HITSZ</u>	<u>0.777</u>	<u>0.785</u>	<u>0.751</u>	<u>0.771</u>
THUIR	0.739	0.742	0.733	0.738
SJTU	0.772	0.724	0.745	0.747
WHU	0.783	0.770	0.760	0.771
Our approach	0.816	0.801	0.786	0.801
Error Reduction	0.152	0.072	0.110	0.131

Table.3 Performance compares with NLP&CC 2013 CLOA evaluation results

To further investigate the effectiveness of our method, the third set of experiments evaluate the negative transfer detection (NTD) compared to co-training (CO) without negative transfer detection as shown in Table.4 and Fig.3 Here, we use the union of Train_CHN and Train_ENG as labeled data and Dev_CHN as unlabeled data to be transferred in the learning algorithms.

		DVD	Book	Music	Mean
NTD	Best case	0.816	0.801	0.786	0.801
	Best period	0.809	0.798	0.782	0.796
	Mean	0.805	0.795	0.781	0.794
CO	Best case	0.804	0.796	0.783	0.794
	Best period	0.803	0.794	0.781	0.792
	Mean	0.797	0.790	0.775	0.787

Table.4 CLOA performances

Taking all categories of data, our proposed method improves the overall average precision (the best cases) from 79.4% to 80.1% when compared to the state of the art system which translates to error reduction of 3.40% (p-value \leq 0.01 in Wilcoxon signed rank test). Although the improvement does not seem large, our

¹<http://tcci.ccf.org.cn/conference/2013/dldoc/evdata03.zip>

²<http://tcci.ccf.org.cn/conference/2013/dldoc/evres03.pdf>

³<https://translate.google.com>

⁴<http://svmlight.joachims.org/>

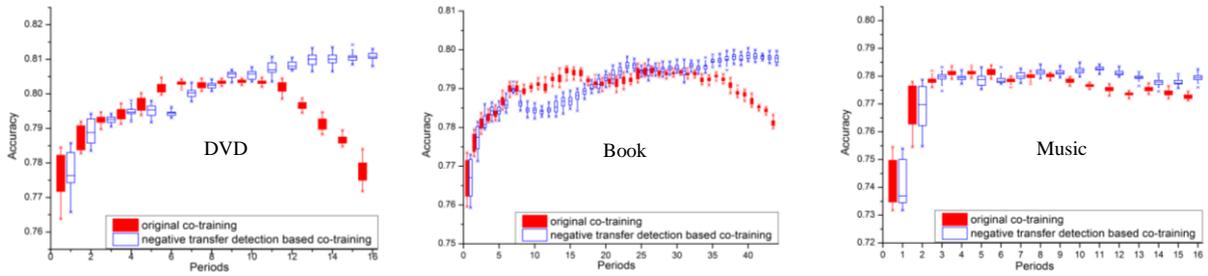


Figure 3 Performance of negative transfer detection vs. co-training

algorithm shows a different behavior in that it can continue to make use of available training data to improve the system performance. In other words, we do not need to identify the tipping point where the performance degradation can occur when more training samples are used. Our approach has also shown the advantage of stable improvement.

In the most practical tasks, co-training based approach has the difficulty to determine when to stop the training process because of the negative transfer. And thus, there is no sure way to obtain the above best average precision. On the contrary, the performance of our proposed approach keeps stable improvement with more iterations, i.e. our approach has a much better chance to ensure the best performance. Another experiment is conducted to compare the performance of our proposed transfer learning based approach with supervised learning. Here, the achieved performance of 3-folder cross validation are given in Table 5.

	DVD	Book	Music	Average
Supervised	0.833	0.800	0.801	0.811
Our approach	0.816	0.801	0.786	0.801

Table.5 Comparison with supervised learning

The accuracy of our approach is only 1.0% lower than the supervised learning using 2/3 of Test_CHN. In the BOOK subset, our approach achieves match result. Note that the performance gap in different subsets shows positive correlation to the size of Dev_CHN. The more samples are given in Dev_CHN, a higher precision is achieved even though these samples are unlabeled. According to the theorem of PAC, we know that the accuracy of a classifier training from a large enough training set with confined class noise rate will approximate the accuracy of classifier training from a non-class noise training set. This experiment shows that our proposed negative transfer detection controls the class noise rate in a very limited boundary. Theoretically speaking, it can catch up with the performance of supervised learning if enough unlabeled samples are available. In fact, such an advantage is the essence of our proposed approach.

cally speaking, it can catch up with the performance of supervised learning if enough unlabeled samples are available. In fact, such an advantage is the essence of our proposed approach.

5 Conclusion

In this paper, we propose a negative transfer detection approach for transfer learning method in order to handle cumulative class noise and reduce negative transfer in the process of transfer learning. The basic idea is to utilize high quality samples after transfer learning to detect class noise in transferred samples. We take cross lingual opinion analysis as the data set to evaluate our method. Experiments show that our proposed approach obtains a more stable performance improvement by reducing negative transfers. Our approach reduced 13.1% errors than the top system on the NLP&CC 2013 CLOA evaluation dataset. In BOOK category it even achieves better result than the supervised learning. Experimental results also show that our approach can obtain better performance when the transferred samples are added incrementally, which in previous works would decrease the system performance. In future work, we plan to extend this method into other language/domain resources to identify more transferred samples.

Acknowledgement

This research is supported by NSFC 61203378, 61300112, 61370165, Natural Science Foundation of Guangdong S2013010014475, MOE Specialized Research Fund for the Doctoral Program of Higher Education 20122302120070, Open Projects Program of National Laboratory of Pattern Recognition, Shenzhen Foundational Research Funding JCYJ20120613152557576, JC201005260118A, Shenzhen International Cooperation Research Funding GJHZ20120613110641217 and Hong Kong Polytechnic University Project code Z0EP.

Reference

- Angluin, D., Laird, P. 1988. Learning from Noisy Examples. *Machine Learning*, 2(4): 343-370.
- Arnold, A., Nallapati, R., Cohen, W. W. 2007. A Comparative Study of Methods for Transductive Transfer Learning. In Proc. 7th IEEE ICDM Workshops, pages 77-82.
- Aue, A., Gamon, M. 2005. Customizing Sentiment Classifiers to New Domains: a Case Study, In Proc. of t RANLP.
- Blitzer, J., McDonald, R., Pereira, F. 2006. Domain Adaptation with Structural Correspondence Learning. In Proc. EMNLP, 120-128.
- Brodley, C. E., Friedl, M. A. 1999. Identifying and Eliminating Mislabeled Training Instances. *Journal of Artificial Intelligence Research*, 11:131-167.
- Chao, D., Guo, M. Z., Liu, Y., Li, H. F. 2008. Participatory Learning based Semi-supervised Classification. In Proc. of 4th ICNC, pages 207-216.
- Cheng, Y., Li, Q. Y. 2009. Transfer Learning with Data Edit. *LNAI*, pages 427-434.
- Chen, M., Weinberger, K. Q., Blitzer, J. C. 2011. Co-Training for Domain Adaptation. In Proc. of 23th NIPS.
- Fukumoto, F., Suzuki, Y., Matsuyoshi, S. 2013. Text Classification from Positive and Unlabeled Data using Misclassified Data Correction. In Proc. of 51st ACL, pages 474-478.
- Gui, L., Xu, R., Xu, J., et al. 2013. A Mixed Model for Cross Lingual Opinion Analysis. In CCIS, 400, pages 93-104.
- Huang, J., Smola, A., Gretton, A., Borgwardt, K.M., Scholkopf, B. 2007. Correcting Sample Selection Bias by Unlabeled Data. In Proc. of 19th NIPS, pages 601-608.
- Jiang, Y., Zhou, Z. H. 2004. Editing Training Data for kNN Classifiers with Neural Network Ensemble. In LNCS, 3173, pages 356-361.
- Li, M., Zhou, Z. H. 2005. SETRED: Self-Training with Editing. In Proc. of PAKDD, pages 611-621.
- Li, M., Zhou, Z. H. 2011. COTRADE: Confident Co-Training With Data Editing. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, 41(6):1612-1627.
- Lu, B., Tang, C. H., Cardie, C., Tsou, B. K. 2011. Joint Bilingual Sentiment Classification with Unlabeled Parallel Corpora. In Proc. of 49th ACL, pages 320-330.
- Meng, X. F., Wei, F. R., Liu, X. H., et al. 2012. Cross-Lingual Mixture Model for Sentiment Classification. In Proc. of 50th ACL, pages 572-581.
- Muhlenbach, F., Lallich, S., Zighed, D. A. 2004. Identifying and Handling Mislabeled Instances. *Journal of Intelligent Information System*, 22(1): 89-109.
- Pan, S. J., Yang, Q. 2010. A Survey on Transfer Learning, *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345-1360.
- Sindhwani, V., Rosenberg, D. S. 2008. An RKHS for Multi-view Learning and Manifold Co-Regularization. In Proc. of 25th ICML, pages 976-983.
- Sluban, B., Gamberger, D., Lavra, N. 2010. Advances in Class Noise Detection. In Proc. 19th ECAI, pages 1105-1106.
- Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P.V., Kawanabe, M. 2008. Direct Importance Estimation with Model Selection and its Application to Covariate Shift Adaptation. In Proc. 20th NIPS.
- Wan, X. 2009. Co-Training for Cross-Lingual Sentiment Classification, In Proc. of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP, 235-243.
- Zhang, H. P., Yu, H. K., Xiong, D. Y., and Liu., Q. 2003. HHMM-based Chinese Lexical Analyzer ICTCLAS. In 2nd SIGHAN workshop affiliated with 41th ACL, pages 184-187.
- Zhou, X., Wan X., Xiao, J. 2011. Cross-Language Opinion Target Extraction in Review Texts. In Proc. of IEEE 12th ICDM, pages 1200-1205.
- Zhu, X. Q., Wu, X. D., Chen, Q. J. 2003. Eliminating Class Noise in Large Datasets. In Proc. of 12th ICML, pages 920-927.
- Zhu, X. Q. 2004. Cost-guided Class Noise Handling for Effective Cost-sensitive Learning In Proc. of 4th IEEE ICDM, pages 297-304.
- Zighed, D. A., Lallich, S., Muhlenbach, F. 2002. Separability Index in Supervised Learning. In Proc. of PKDD, pages 475-487.

Author Index

- Agirre, Eneko, 649
Aharoni, Roece, 289
Al Tarouti, Feras, 106
Alamir-Salloum, Linda, 772
Aletras, Nikolaos, 631
Alhelbawy, Ayman, 75
Andreas, Jacob, 822
Auli, Michael, 136
- Baldwin, Timothy, 93
Bamman, David, 828
Banchs, Rafael E., 19
Bansal, Mohit, 809
Baytin, Alexey, 168
Beigman Klebanov, Beata, 247, 390
Beigman, Eyal, 390
Beller, Charley, 181
Beloucif, Meriem, 765
Ben-Ami, Zvi, 87
Bengoetxea, Kepa, 649
Berg-Kirkpatrick, Taylor, 118
Bergsma, Shane, 181
Bethard, Steven, 81, 501
Bhattacharyya, Pushpak, 36
Bies, Ann, 668
Blunsom, Phil, 224
Bonial, Claire, 397
Burstein, Jill, 247
- Cahill, Aoife, 174
Cai, Jingsheng, 155
Cano Basave, Amparo Elizabeth, 618
Cardie, Claire, 693
Carpuat, Marine, 283
Cassidy, Taylor, 501
Chae, Heemoon, 637
Chai, Joyce Y., 13
Chakraborti, Sutanu, 55
Chambers, Nathanael, 501
Charniak, Eugene, 512, 592
Charton, Eric, 476
Chen, Li, 345
Chen, Liangyu, 700
Chen, Yanqing, 383
- Choe, Do Kook, 512
Chow, Kam-Pui, 421
Chrupała, Grzegorz, 680
Ciobanu, Alina Maria, 99
Clark, Stephen, 835
Clemmer, Alex, 446
Cohn, Trevor, 150
Cotterell, Ryan, 625
Crasborn, Onno, 370
- Dagan, Ido, 739
Daxenberger, Johannes, 187
DeNero, John, 816
Deng, Zhihong, 218
Diab, Mona, 772
Dinu, Liviu P., 99
Dligach, Dmitriy, 81
Dong, Li, 49
Dredze, Mark, 545, 674
Drude, Sebastian, 370
Duh, Kevin, 143
Duma, Daniel, 358
Dunlop, Aaron, 797
Dunn, Jonathan, 745
Dupoux, Emmanuel, 1
Dyer, Chris, 828
- Ehsani, Razieh, 112
Eisenstein, Jacob, 265, 415, 538
Eisner, Jason, 625
Elfardy, Heba, 772
Elliott, Desmond, 452
Elsner, Micha, 265
Elson, David, 230
- Fang, Rui, 13
Farra, Noura, 161
Feldman, Ronen, 87
Finch, Andrew, 752
Fine, Alex B., 7
Fourtassi, Abdellah, 1
Frank, Austin F., 7
Friedrich, Annemarie, 517
Fukumoto, Fumiyo, 241

Gagnon, Michel, 476
Gaizauskas, Robert, 75
Galinskaya, Irina, 168
Gao, Jianfeng, 136
Gawron, Jean Mark, 296
Gebre, Binyam Gebrekidan, 370
Gelling, Douwe, 150
Gilbert, Eric, 415
Gildea, Daniel, 236
Gimpel, Kevin, 809
Gojenola, Koldo, 649
Goldberg, Yoav, 289, 302
Goldwater, Sharon, 265
Görgün, Onur, 112
Green, Spence, 206
Grigonyte, Gintare, 93
Grishman, Ralph, 68, 732
Gubanov, Sergey, 168
Guerini, Marco, 427
Gui, Lin, 860
Gurevych, Iryna, 187

Habash, Nizar, 161, 772
Hachey, Ben, 464
Han, Jiawei, 706
Han, Xianpei, 61, 718
Harman, Craig, 181
Hartshorne, Joshua K., 397
He, Xiaodong, 643
He, Yulan, 618, 700
Heafield, Kenneth, 130
Hearst, Marti A., 272
Heilman, Michael, 174
Hermann, Karl Moritz, 224
Heskes, Tom, 370
Hieber, Felix, 488
Hill, Felix, 725, 835
Hingmire, Swapnil, 55
Hirao, Tsutomu, 315
Hirst, Graeme, 531
Hong, Yu, 569
Horn, Colby, 458
Hovy, Dirk, 377, 482, 507
Hovy, Eduard, 24, 30
Huang, Hongzhao, 706
Huang, Liang, 785
Huang, Yu-Yang, 611
Hwa, Rebecca, 599
Hwang, Seung-won, 848

Ichikawa, Hiroshi, 557
Ittycheriah, Abe, 785

Jaeger, T. Florian, 7
Jean-Louis, Ludovic, 476
Ji, Heng, 278, 495, 706
Jochim, Charles, 42
Joshi, Aditya, 36

Kalchbrenner, Nal, 212
Kalita, Jugal, 106
Kartsaklis, Dimitri, 212
Kauchak, David, 458
Kawahara, Daisuke, 253
Kayser, Michael, 130
Kazawa, Hideto, 557
Kazemi, Mohammad, 236
Keller, Frank, 452
Kertz, Laura, 512
Kho, Alvin, 81
Kiela, Douwe, 835
Kikuchi, Yuta, 315
Kim, Seokhwan, 19
Kim, Young-Bum, 637
Kim, Yu-Seop, 637
Klein, Dan, 118, 822
Klein, Ewan, 358
Knight, Kevin, 278, 706
Knowles, Rebecca, 181
Kočíský, Tomáš, 224
Koehn, Philipp, 574
Köhn, Arne, 803
Kondrak, Grzegorz, 854
Koppel, Moshe, 289
Korhonen, Anna, 725, 835
Kozhevnikov, Mikhail, 579
Kroch, Anthony, 662, 668
Kudo, Taku, 557
Kulick, Seth, 662, 668
Kuo, Tsung-Ting, 611
Kurimo, Mikko, 259
Kurohashi, Sadao, 253

Lall, Ashwin, 687
Lam, Khang Nhut, 106
Lavaee, Rahman, 236
Lee, Lillian, 403
Lee, Taesung, 848
Levitan, Rivka, 230
Levy, Omer, 302
Li, Haizhou, 19
Li, Hao, 495
Li, Junyi Jessy, 283
Li, Shoushan, 842
Li, Si, 199

Lieberman, Mark, 668
Lin, Chen, 81
Lin, Shou-De, 611
Lindén, Krister, 259
Liu, Bin, 860
Liu, Bing, 345
Liu, Changsong, 13
Liu, Fei, 605
Liu, Hao, 569
Liu, Hongxiao, 253
Liu, Le, 569
Liu, Wei, 495
Liu, Yang, 327
Livescu, Karen, 809
Lo, Chi-kiu, 765
Lopez, Melissa, 174
Louis, Annie, 333
Lu, Qin, 860
Lu, Wen-Hsiang, 470
Luo, Xiaoqiang, 24, 30

Ma, Ji, 791
Madnani, Nitin, 174, 247
Manduca, Cathryn, 458
Manning, Christopher D., 124, 130, 193, 206
Mason, Rebecca, 592
May, Chandler, 446
Mazidi, Karen, 321
McDonald, Ryan, 656
McDowell, Bill, 501
Meek, Christopher, 643
Menzel, Wolfgang, 803
Meurs, Marie-Jean, 476
Mi, Haitao, 785
Mihalcea, Rada, 440
Miller, Timothy, 81
Min, Bonan, 732
Mishra, Abhijit, 36
Mitchell, Margaret, 181
Mitra, Tanushree, 415
Monroe, Will, 206
Mott, Justin, 668
Mulholland, Matthew, 174
Muralidharan, Aditi, 272

Nagata, Masaaki, 315
Naim, Iftekhar, 236
Nakamura, Satoshi, 551
Nenkova, Ani, 283
Neubig, Graham, 143, 551
Ney, Hermann, 759
Ng, Vincent, 30

Nguyen, Thien Huu, 68
Nicolai, Garrett, 854
Nielsen, Rodney D., 321
Nivre, Joakim, 649
Nothman, Joel, 464
Nuhn, Malte, 759

Oberlander, Jon, 409
Oda, Yusuke, 551
Okumura, Manabu, 315
Osborne, Miles, 687
Özbal, Gözde, 352

Palmer, Alexis, 517
Palmer, Martha, 397
Pan, Xiaoman, 706
Peng, Nanyun, 625, 674
Peng, Zhiyong, 345
Perek, Florent, 309
Pérez-Rosas, Verónica, 440
Pershina, Maria, 732
Pighin, Daniele, 352
Plank, Barbara, 377, 507
Prabhakaran, Vinodkumar, 339
Pradhan, Sameer, 24, 30, 81

Qian, Tieyun, 345
Qian, Xian, 327

Radford, Will, 464
Ramanath, Rohan, 605
Rambow, Owen, 339
Recasens, Marta, 24, 30
Riezler, Stefan, 488
Roark, Brian, 364, 797
Rosenfeld, Binyamin, 87
Roth, Michael, 524
Rozovskaya, Alla, 161
Ruokolainen, Teemu, 259

Sadeh, Norman, 605
Sadrzadeh, Mehrnoosh, 212
Saers, Markus, 765
Saint-Amand, Herve, 574
Sakti, Sakriani, 551
Salloum, Wael, 772
Salway, Andrew, 712
Santorini, Beatrice, 662, 668
Savova, Guergana, 81
Schamoni, Shigehiko, 488
Schatz, Thomas, 1
Schulte im Walde, Sabine, 524
Schütze, Hinrich, 42

Senthamilselvan, Nivvedan, 36
Søgaard, Anders, 377, 507
She, Lanbo, 13
Shen, Mo, 253
Shi, Xing, 278
Silfverberg, Miikka, 259
Sirts, Kairit, 265
Skiena, Steven, 383
Smith, Noah A., 605, 828
Snyder, Benjamin, 637
Sokolov, Artem, 488
Solak, Ercan, 112
Somasundaran, Swapna, 247
Soni, Sandeep, 415
Sproat, Richard, 364
Staiano, Jacopo, 427
Stern, Asher, 739
Stevenson, Mark, 631
Strapparava, Carlo, 352
Strube, Michael, 30
Sumita, Eiichiro, 155, 752
Sun, Le, 61, 718
Sun, Yizhou, 706
Suzuki, Yoshimi, 241

Takamura, Hiroya, 315
Tan, Chenhao, 403
Tan, Chuanqi, 49
Tang, Duyu, 49
Tetreault, Joel, 174
Tibshirani, Julie, 124
Titov, Ivan, 579
Toda, Tomoki, 551
Tomeh, Nadi, 161
Touileb, Samia, 712
Tsai, Kun-Yu, 470
Tsai, Richard Tzong-Han, 586
Tsoukala, Chara, 574

Utiyama, Masao, 155, 752

Van Durme, Benjamin, 7, 181, 446, 687
Varadarajan, Balakrishnan, 1
Voigt, Rob, 193

Wallace, Byron C., 512
Wang, Lu, 693
Wang, Mengqiu, 193
Wang, Ting-Xuan, 470
Wang, Tong, 531
Wang, Xiaolin, 752
Wang, Xiaolong, 860
Wang, Xing, 569

Wang, Yiming, 674
Wang, Yu-Chun, 586
Wei, Furu, 49
Wen, Zhen, 706
Wilson, Shomir, 409
Wittenburg, Peter, 370
Wu, Chun-Kai, 586
Wu, Dekai, 765

Xia, Rui, 842
Xiang, Bing, 434
Xiao, Tong, 563
Xu, Jian, 860
Xu, Jun, 860
Xu, Ke, 49
Xu, Ruifeng, 618, 860
Xu, Wei, 732
Xue, Huichao, 599
Xue, Nianwen, 199

Yıldız, Olcay Taner, 112
Yan, Rui, 611
Yang, Min, 421
Yang, Yi, 538
Yang, Yunlun, 218
Yao, Jianmin, 569
Yarmohammadi, Mahsa, 797
Yener, Bulent, 706
Yih, Wen-tau, 643
Yu, Hongliang, 218
Yu, Mo, 545

Zhai, Feifei, 779
Zhang, Boliang, 706
Zhang, Chunliang, 563
Zhang, Hao, 656
Zhang, Hui, 816
Zhang, Jiajun, 779
Zhang, Yue, 649, 791
Zhang, Yujie, 155
Zhao, Kai, 785
Zhou, Deyu, 700
Zhou, Guodong, 842
Zhou, Liang, 434
Zhou, Ming, 49
Zhou, Yu, 779
Zhu, Dingju, 421
Zhu, Jingbo, 563, 791
Zhu, Zhu, 842
Zong, Chengqing, 779