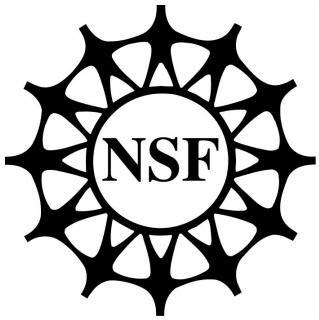
ACL-08: HLT

46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies

Proceedings of the Student Research Workshop

June 16, 2008
The Ohio State University
Columbus, Ohio, USA

Production and Manufacturing by
Omnipress Inc.
2600 Anderson Street
Madison, WI 53707
USA



Sponsored by
The National Science Foundation

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

Introduction

Welcome to the ACL-08: HLT Student Research Workshop! The Student Research Workshop is now an established tradition at ACL conferences and provides a venue for student researchers investigating topics in Computational Linguistics and Natural Language Processing to present their work and receive feedback. New this year was encouraging researchers in Information Retrieval to be involved, both reviewers and authors. We received a total of 27 submissions coming from 11 different countries, and accepted 12 of them. 5 will be presented orally, and 7 as posters, during a common poster session with the main conference. A total of 61 students and senior researchers agreed to serve on the program committee, which allowed us to assign 5 reviewers per paper. We would like to thank the reviewers for understanding the spirit of the Student Research Workshop and giving careful and constructive reviews. We hope their comments will be helpful to all the students who submitted their work. All presenters received financial support from the U.S. National Science Foundation to assist them in their travel to Columbus.

We are very grateful to Jan Wiebe, our faculty advisor, for her advice, constant support, and obtaining funding. Finally, we would like to thank the general chair of ACL-08: HLT, Kathleen McKeown, the program chairs, Johanna Moore, Simone Teufel, James Allan, and Sadaoki Furui, the publications chairs Joakim Nivre and Noah Smith, Chris Brew and the local organization committee, and Priscilla Rasmussen.

The ACL-08: HLT Student Research Workshop co-chairs:
Ebru Arısoy, Keisuke Inoue and Wolfgang Maier

Co-chairs:

Ebru Arısoy, Boğaziçi University, Turkey
Wolfgang Maier, University of Tübingen, Germany
Keisuke Inoue, University of Syracuse, USA

Faculty advisor:

Jan Wiebe, University of Pittsburgh, USA

Program committee:

Murat Akbacak, SRI International, USA
Tanel Alumäe, Tallinn University of Technology, Estonia
Michiel Bacchiani, Google Inc., USA
Timothy Baldwin, University of Melbourne, Australia
Chris Bartels, University of Washington, USA
Tilman Becker, DFKI Saarbrücken, Germany
Marine Carpuat, Hong Kong University of Science and Technology, Hong Kong
Gülşen Cebiroğlu Eryiğit, Istanbul Technical University, Turkey
Özlem Çetinoğlu, Sabancı University, Turkey
Mathias Creutz, Nokia Research Center, Finland
Montse Cuadros, Polytechnic University of Catalonia, Spain
Anne Diekema, Syracuse University, USA
Markus Dreyer, Johns Hopkins University, USA
Kevin Duh, University of Washington, USA
Koji Eguchi, Kobe University, Japan
Hakan Erdoğan, Sabancı University, Turkey
Katja Filippova, EML Research, Germany
Seeger Fisher, OGI School of Science and Engineering, USA
Dilek Hakkani-Tür, ICSI, USA
Dustin Hillard, University of Washington, USA
Teemu Hirsimäki, Helsinki University of Technology, Finland
Tatsuya Kawahara, Kyoto University, Japan
Eric Kow, University of Brighton, UK
Sandra Kübler, Indiana University, USA
Giridhar Kumaran, University of Massachusetts Amherst, USA
Mikko Kurimo, Helsinki University of Technology, Finland
Staffan Larsson, Göteborg University, Sweden
Lin-Shan Lee, National Taiwan University, Taiwan
Kyung-Soon Lee, Chonbuk National University, Korea
Timm Lichte, University of Tübingen, Germany
Andrej Ljolje, AT&T Labs - Research, USA

Berenike Loos, German National Library, Germany
Robert Luk, The Hong Kong Polytechnic University, Hong Kong
Lambert Mathias, Johns Hopkins University, USA
Olena Medelyan, University of Waikato, New Zealand
Quiozhu Mei, University of Illinois at Urbana-Champaign, USA
Simon Mille, Pompeu Fabra University, Spain
Mathias Möhl, Saarland University, Germany
Kemal Oflazer, Sabancı University, Turkey
Paul Ogilvie, mSpoke, Inc., USA
Constantin Orasan, University of Wolverhampton, UK
Yannick Parmentier, University of Tübingen, Germany
Thomas Pellegrini, LIMSI, France
Adam Przepiórkowski, Institute of Computer Science, Polish Academy of Sciences, Poland
Janne Pylkkönen, Helsinki University of Technology, Finland
Georg Rehm, University of Tübingen, Germany
Brian Roark, OGI School of Science and Engineering, USA
Haşim Sak, Boğaziçi University, Turkey
Murat Saraçlar, Boğaziçi University, Turkey
Ruhi Sarikaya, IBM Watson Research Center, USA
Oliver Schonefeld, University of Bielefeld, Germany
Izak Shafran, OGI School of Science and Engineering, USA
Anders Søgaard, University of Potsdam, Germany
Richard Sproat, University of Illinois at Urbana-Champaign, USA
Yael Sygal, University of Haifa, Israel
Cüneyd Tantuğ, Istanbul Technical University, Turkey
Gökhan Tür, SRI International, USA
Suzan Verberne, University of Nijmegen, The Netherlands
Ellen Voorhees, National Institute of Standards and Technology, USA
Christopher White, Johns Hopkins University, USA
Hans Friedrich Witschel, University of Leipzig, Germany

Table of Contents

Workshop Program

Monday, June 16, 2008

Oral Session

3:45–4:10 *A Supervised Learning Approach to Automatic Synonym Identification Based on Distributional Features*
Masato Hagiwara

4:10–4:35 *An Integraged Architecture for Generating Parenthetical Constructions*
Eva Banik

4:35–5:00 *Inferring Activity Time in News through Event Modeling*
Vladimir Eidelman

5:00–5:25 *Combining Source and Target Language Information for Name Tagging of Machine Translation Output*
Shasha Liao

5:25–5:50 *A Re-examination on Features in Regression Based Approach to Automatic MT Evaluation*
Shuqi Sun, Yin Chen and Jufeng Li

Poster Session

6:00–8:30 *The Role of Positive Feedback in Intelligent Tutoring Systems*
Davide Fossati

Arabic Language Modeling with Finite State Transducers
Ilana Heintz

Impact of Initiative on Collaborative Problem Solving
Cynthia Kersey

An Unsupervised Vector Approach to Biomedical Term Disambiguation: Integrating UMLS and Medline
Bridget McInnes

A Subcategorization Acquisition System for French Verbs
Cédric Messiant

Adaptive Language Modeling for Word Prediction
Keith Trnka

A Hierarchical Approach to Encoding Medical Concepts for Clinical Notes
Yitao Zhang

A Supervised Learning Approach to Automatic Synonym Identification based on Distributional Features

Masato Hagiwara
Graduate School of Information Science
Nagoya University
Furo-cho, Chikusa-ku, Nagoya 464-8603, JAPAN
hagiwara@kl.i.is.nagoya-u.ac.jp

Abstract

Distributional similarity has been widely used to capture the semantic relatedness of words in many NLP tasks. However, various parameters such as similarity measures must be hand-tuned to make it work effectively. Instead, we propose a novel approach to synonym identification based on supervised learning and *distributional features*, which correspond to the commonality of individual context types shared by word pairs. Considering the integration with *pattern-based features*, we have built and compared five synonym classifiers. The evaluation experiment has shown a dramatic performance increase of over 120% on the F-1 measure basis, compared to the conventional similarity-based classification. On the other hand, the pattern-based features have appeared almost redundant.

1 Introduction

Semantic similarity of words is one of the most important lexical knowledge for NLP tasks including word sense disambiguation and automatic thesaurus construction. To measure the semantic relatedness of words, a concept called *distributional similarity* has been widely used. Distributional similarity represents the relatedness of two words by the commonality of contexts the words share, based on the *distributional hypothesis* (Harris, 1985), which states that semantically similar words share similar contexts.

A number of researches which utilized distributional similarity have been conducted, including (Hindle, 1990; Lin, 1998; Geffet and Dagan, 2004) and many others. Although they have been successful in acquiring related words, various parameters such as similarity measures and weighting are involved. As Weeds et al. (2004) pointed out, "it is not at all obvious that one universally best measure exists for all application," thus they must be tuned by hand in an ad-hoc manner. The fact that no theoretic basis is given is making the matter more difficult.

On the other hand, if we pay attention to lexical knowledge acquisition in general, a variety of systems which utilized *syntactic patterns* are found in the literature. In her landmark paper in the field, Hearst (1992) utilized syntactic patterns such as "such X as Y" and "Y and other X," and extracted hypernym/hyponym relation of X and Y. Roark and Charniak (1998) applied this idea to extraction of words which belong to the same categories, utilizing syntactic relations such as conjunctions and appositives. What is worth attention here is that supervised machine learning is easily incorporated with syntactic patterns. For example, Snow et al. (2004) further extended Hearst's idea and built hypernym classifiers based on machine learning and syntactic pattern-based features, with a considerable success.

These two independent approaches, distributional similarity and syntactic patterns, were finally integrated by Mirkin et al. (2006). Although they reported that their system successfully improved the performance, it did not achieve a complete integration and was still relying on an independent module to compute the similarity. This configuration inherits a large portion of drawbacks of the similarity-based approach mentioned above. To achieve a full integration of both approaches, we suppose that re-

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 1–6,
Columbus, June 2008.

formalization of similarity-based approach would be essential, as pattern-based approach is enhanced with the supervised machine learning.

In this paper, we propose a novel approach to automatic synonym identification based on supervised learning technique. Firstly, we re-formalize synonym acquisition as a classification problem: one which classifies *word pairs* into synonym/non-synonym classes, without depending on a single value of distributional similarity. Instead, classification is done using a set of *distributional features*, which correspond to the degree of commonality of individual context types shared by word pairs. This formalization also enables to incorporate pattern-based features, and we finally build five classifiers based on distributional and/or pattern-based features. In the experiment, their performances are compared in terms of synonym acquisition precision and recall, and the differences of actually acquired synonyms are to be clarified.

The rest of this paper is organized as follows: in Sections 2 and 3, distributional and pattern-based features are defined, along with the extraction methods. Using the features, in Section 4 we build five types of synonym classifiers, and compare their performances in Section 5. Section 6 concludes this paper, mentioning the future direction of this study.

2 Distributional Features

In this section, we firstly describe how we extract contexts from corpora and then how distributional features are constructed for word pairs.

2.1 Context Extraction

We adopted dependency structure as the context of words since it is the most widely used and well-performing contextual information in the past studies (Ruge, 1997; Lin, 1998). In this paper the sophisticated parser RASP Toolkit 2 (Briscoe et al., 2006) was utilized to extract this kind of word relations. We use the following example for illustration purposes: *The library has a large collection of classic books by such authors as Herrick and Shakespeare.* RASP outputs the extracted dependency structure as n-ary relations as follows:

```
(ncsubj have library _)
(dobj have collection)
(det collection a)
(ncmod _ collection large)
(iobj collection of)
(dobj of book)
(ncmod _ book by)
(dobj by author)
(det author such)
(ncmod _ author as)
... ,
```

whose graphical representation is shown in Figure 1.

While the RASP outputs are n-ary relations in general, what we need here is co-occurrences of words and contexts, so we extract the set of co-occurrences of stemmed words and contexts by taking out the target word from the relation and replacing the slot by an asterisk "*":

```
library     - (ncsubj have * _)
library     - (det * The)
collection  - (dobj have *)
collection  - (det * a)
collection  - (ncmod _ * large)
collection  - (iobj * of)
book        - (dobj of *)
book        - (ncmod _ * by)
book        - (ncmod _ * classic)
author      - (dobj by *)
author      - (det * such)
...
```

Summing all these up produces the raw co-occurrence count $N(w, c)$ of the word w and the context c. In the following, the set of context types co-occurring with the word w is denoted as $C(w)$, i.e., $C(w) = \{c|N(w, c) > 0\}$.

2.2 Feature Construction

Using the co-occurrences extracted above, we define distributional features $f_j^D(w_1, w_2)$ for the word pair (w_1, w_2). The feature value f_j^D is determined so that it represents the degree of commonality of the context c_j shared by the word pair. We adopted *pointwise total correlation*, one of the generalizations of pointwise mutual information, as the feature value:

$$f_j^D(w_1, w_2) = \log \frac{P(w_1, w_2, c_j)}{P(w_1)P(w_2)P(c_j)}. \quad (1)$$

The advantage of this feature construction is that, given the independence assumption between the words w_1 and w_2, the feature value is easily calculated as the simple sum of two corresponding pointwise mutual information weights as:

$$f_j^D(w_1, w_2) = \mathrm{PMI}(w_1, c_j) + \mathrm{PMI}(w_2, c_j), \quad (2)$$

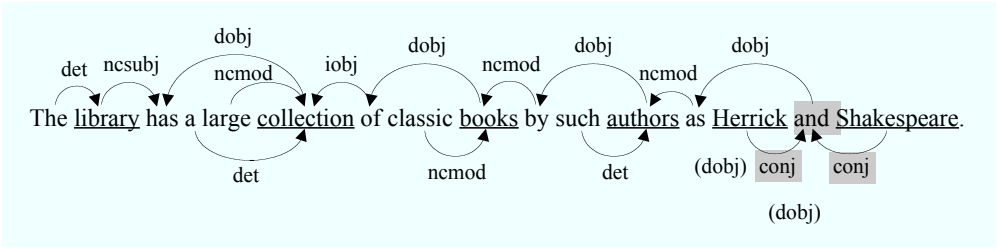


Figure 1: Dependency structure of the example sentence, along with conjunction shortcuts (dotted lines).

where the value of PMI, which is also the weights $\mathrm{wgt}(w_i, c_j)$ assigned for distributional similarity, is calculated as:

$$\mathrm{wgt}(w_i, c_j) = \mathrm{PMI}(w_i, c_j) = \log \frac{P(w_i, c_j)}{P(w_i)P(c_j)}. \quad (3)$$

There are three things to note here: when $N(w_i, c_j) = 0$ and PMI cannot be defined, then we define $\mathrm{wgt}(w_i, c_j) = 0$. Also, because it has been shown (Curran and Moens, 2002) that negative PMI values worsen the distributional similarity performance, we bound PMI so that $\mathrm{wgt}(w_i, c_j) = 0$ if $\mathrm{PMI}(w_i, c_j) < 0$. Finally, the feature value $f_j^D(w_1, w_2)$ is defined as shown in Equation (2) only when the context c_j co-occurs with both w_1 and w_2. In other words, $f_j^D(w_1, w_2) = 0$ if $\mathrm{PMI}(w_1, c_j) = 0$ and/or $\mathrm{PMI}(w_2, c_j) = 0$.

3 Pattern-based Features

This section describes the other type of features, extracted from syntactic patterns in sentences.

3.1 Syntactic Pattern Extraction

We define *syntactic patterns* based on dependency structure of sentences. Following Snow et al. (2004)'s definition, the syntactic pattern of words w_1, w_2 is defined as the concatenation of the words and relations which are on the dependency path from w_1 to w_2, not including w_1 and w_2 themselves.

The syntactic pattern of word *authors* and *books* in Figure 1 is, for example, `dobj:by:ncmod`, while that of *authors* and *Herrick* is `ncmod-of:as:dobj-of:and:conj-of`. Notice that, although not shown in the figure, every relation has a reverse edge as its counterpart, with the direction opposite and the postfix "`-of`" attached to the label. This allows to follow the relations in reverse, increasing the flexibility and expressive power of patterns.

In the experiment, we limited the maximum length of syntactic path to five, meaning that word pairs having six or more relations in between were disregarded. Also, we considered *conjunction shortcuts* to capture the lexical relations more precisely, following Snow et al. (2004). This modification cuts short the `conj` edges when nouns are connected by conjunctions such as *and* and *or*. After this shortcut, the syntactic pattern between *authors* and *Herrick* is `ncmod-of:as:dobj-of`, and that of *Herrick* and *Shakespeare* is `conj-and`, which is a newly introduced special symmetric relation, indicating that the nouns are mutually conjunctional.

3.2 Feature Construction

After the corpus is analyzed and patterns are extracted, the pattern based feature $f_k^P(w_1, w_2)$, which corresponds to the syntactic pattern p_k, is defined as the conditional probability of observing p_k given that the pair (w_1, w_2) is observed. This definition is similar to (Mirkin et al., 2006) and is calculated as:

$$f_k^P(w_1, w_2) = P(p_k | w_1, w_2) = \frac{N(w_1, w_2, p_k)}{N(w_1, w_2)}. \quad (4)$$

4 Synonym Classifiers

Now that we have all the features to consider, we construct the following five classifiers. This section gives the construction detail of the classifiers and corresponding feature vectors.

Distributional Similarity (DSIM) DSIM classifier is simple acquisition relying only on distributional similarity, not on supervised learning. Similar to conventional methods, distributional similarity between words w_1 and w_2, $\mathrm{sim}(w_1, w_2)$, is calculated for each word pair using Jaccard coefficient:

$$\frac{\sum_{c \in C(w_1) \cap C(w_2)} \min(\mathrm{wgt}(w_1, c), \mathrm{wgt}(w_2, c))}{\sum_{c \in C(w_1) \cup C(w_2)} \max(\mathrm{wgt}(w_1, c), \mathrm{wgt}(w_2, c))},$$

considering the preliminary experimental result. A threshold is set on the similarity and classification is performed based on whether the similarity is above or below of the given threshold. How to optimally set this threshold is described later in Section 5.1.

Distributional Features (DFEAT) DFEAT classifier does not rely on the conventional distributional similarity and instead uses the distributional features described in Section 2. The feature vector $\vec{v}$ of a word pair (w_1, w_2) is constructed as:

$$\vec{v} = (f_1^D,\ ...,\ f_M^D). \tag{5}$$

Pattern-based Features (PAT) This classifier PAT uses only pattern-based features, essentially the same as the classifier of Snow et al. (2004). The feature vector is:

$$\vec{v} = (f_1^P,\ ...,\ f_K^P). \tag{6}$$

Distributional Similarity and Pattern-based Features (DSIM-PAT) DSIM-PAT uses the distributional similarity of pairs as a feature, in addition to pattern-based features. This classifier is essentially the same as the integration method proposed by Mirkin et al. (2006). Letting $f^S = \mathrm{sim}(w_1, w_2)$, the feature vector is:

$$\vec{v} = (f^S,\ f_1^P,\ ...,\ f_K^P). \tag{7}$$

Distributional and Pattern-based Features (DFEAT-PAT) The last classifier, DFEAT-PAT, truly integrates both distributional and pattern-based features. The feature vector is constructed by replacing the f^S component of DSIM-PAT with distributional features $f_1^D,\ ...,\ f_M^D$ as:

$$\vec{v} = (f_1^D,\ ...,\ f_M^D,\ f_1^P,\ ...,\ f_K^P). \tag{8}$$

5 Experiments

Finally, this section describes the experimental setting and the comparison of synonym classifiers.

5.1 Experimental Settings

Corpus and Preprocessing As for the corpus, New York Times section (1994) of English Gigaword [1], consisting of approx. 46,000 documents, 922,000 sentences, and 30 million words, was analyzed to obtain word-context co-occurrences.

This can yield 10,000 or more context types, thus we applied feature selection and reduced the dimensionality. Firstly, we simply applied frequency cutoff to filter out any words and contexts with low frequency. More specifically, any words w such that $\sum_c N(w,c) < \theta_f$ and any contexts c such that $\sum_w N(w,c) < \theta_f$, with $\theta_f = 5$, were removed. DF (document frequency) thresholding is then applied, and context types with the lowest values of DF were removed until 10% of the original contexts were left. We verified through a preliminary experiment that this feature selection keeps the performance loss at minimum. As a result, this process left a total of 8,558 context types, or feature dimensionality.

The feature selection was also applied to pattern-based features to avoid high sparseness — only syntactic patterns which occurred more than or equal to 7 times were used. The number of syntactic pattern types left after this process is 17,964.

Supervised Learning Training and test sets were created as follows: firstly, the nouns listed in the Longman Defining Vocabulary (LDV) [2] were chosen as the target words of classification. Then, all the LDV pairs which co-occur more than or equal to 3 times with any of the syntactic patterns, i.e., $\{(w_1, w_2) | w_1, w_2 \in \mathrm{LDV}, \sum_p N(w_1, w_2, p) \geq 3\}$ were classified into synonym/non-synonym classes as mentioned in Section 5.2. All the positive-marked pair, as well as randomly chosen 1 out of 5 negative-marked pairs, were collected as the *example set* E. This random selection is to avoid extreme bias toward the negative examples. The example set E ended up with 2,148 positive and 13,855 negative examples, with their ratio being approx. 6.45.

The example set E was then divided into five partitions to conduct five-fold cross validation, of which four partitions were used for learning and the one for testing. SVM^{light} was adopted for machine learning, and RBF as the kernel. The parameters, i.e., the similarity threshold of DSIM classifier, gamma parameter of RBF kernel, and the cost-factor j of SVM, i.e., the ratio by which training errors on positive examples outweight errors on negative ones,

[1] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2003T05

[2] http://www.cs.utexas.edu/users/kbarker/working_notes/ldoce-vocab.html

Table 1: Performance comparison of synonym classifiers

Classifier	Precision	Recall	F-1
DSIM	33.13%	49.71%	39.76%
DFEAT	95.25%	82.31%	88.30%
PAT	23.86%	45.17%	31.22%
DSIM-PAT	30.62%	51.34%	38.36%
DFEAT-PAT	95.37%	82.31%	88.36%

were optimized using one of the 5-fold cross validation train-test pair on the basis of F-1 measure. The performance was evaluated for the other four train-test pairs and the average values were recorded.

5.2 Evaluation

To test whether or not a given word pair (w_1, w_2) is a synonym pair, three existing thesauri were consulted: Roget's Thesaurus (Roget, 1995), Collins COBUILD Thesaurus (Collins, 2002), and WordNet (Fellbaum, 1998). The union of synonyms obtained when the head word is looked up as a noun is used as the answer set, except for words marked as "idiom," "informal," "slang" and phrases comprised of two or more words. The pair (w_1, w_2) is marked as synonyms if and only if w_2 is contained in the answer set of w_1, or w_1 is contained in that of w_2.

5.3 Classifier Performance

The performances, i.e., precision, recall, and F-1 measure, of the five classifiers were evaluated and shown in Table 1. First of all, we observed a drastic improvement of DFEAT over DSIM — over 120% increase of F-1 measure. When combined with pattern-based features, DSIM-PAT showed a slight recall increase compared to DSIM, partially reconfirming the favorable integration result of (Mirkin et al., 2006). However, the combination DFEAT-PAT showed little change, meaning that the discriminative ability of DFEAT was so high that pattern-based features were almost redundant. To note, the performance of PAT was the lowest, reflecting the fact that synonym pairs rarely occur in the same sentence, making the identification using only syntactic pattern clues even more difficult.

The reason of the drastic improvement is that, as far as we speculate, the supervised learning may have favorably worked to cause the same effect as automatic feature selection technique. Features with high discriminative power may have been automatically promoted. In the distributional similarity setting, in contrast, the contributions of context types are uniformly fixed. In order to elucidate what is happening in this situation, the investigations on machine learning settings, as well as algorithms other than SVM should be conducted as the future work.

5.4 Acquired Synonyms

In the second part of this experiment, we further investigated what kind of synonyms were actually acquired by the classifiers. The targets are not LDV, but all of 27,501 unique nouns appeared in the corpus, because we cannot rule out the possibility that the high performance seen in the previous experiment was simply due to the rather limited target word settings. The rest of the experimental setting was almost the same as the previous one, except that the construction of training set is rather artificial — we used all of the 18,102 positive LDV pairs and randomly chosen 20,000 negative LDV pairs.

Table 2 lists the acquired synonyms of *video* and *program*. The results of DSIM and DFEAT are ordered by distributional similarity and the value of decision function of SVM, respectively. Notice that because neither word is included in LDV, all the pairs of the query and the words listed in the table are guaranteed to be excluded from the training set.

The result shows the superiority of DFEAT over DSIM. The irrelevant words (marked by "*" by human judgement) seen in the DSIM list are demoted and replaced with more relevant words in the DFEAT list. We observed the same trend for lower ranked words and other query words.

6 Conclusion and Future Direction

In this paper, we proposed a novel approach to automatic synonym identification based on supervised machine learning and distributional features. For this purpose, we re-formalized synonym acquisition as a classification problem, and constructed the features as the total correlation of pairs and contexts. Since this formalization allows to integrate pattern-based features in a seamless way, we built five classifiers based on distributional and/or pattern-based features. The result was promising, achieving more than 120% increase over conventional DSIM classi-

Table 2: Acquired synonyms of *video* and *program*

For query word: *video*

Rank	DSIM	DFEAT
1	computer	computer
2	television	television
3	movie	multimedia
4	film	communication
5	food*	entertainment
6	multimedia	advertisement
7	drug*	food*
8	entertainment	recording
9	music	portrait
10	radio	movie

For query word: *program*

Rank	DSIM	DFEAT
1	system	project
2	plan	system
3	project	unit
4	service	status
5	policy	schedule
6	effort*	organization*
7	bill*	activity*
8	company*	plan
9	operation	scheme
10	organization*	policy

fier. Pattern-based features were partially effective when combined with DSIM whereas with DFEAT they were simply redundant.

The impact of this study is that it makes unnecessary to carefully choose similarity measures such as Jaccard's — instead, features can be directly input to supervised learning right after their construction. There are still a great deal of issues to address as the current approach is only in its infancy. For example, the formalization of distributional features requires further investigation. Although we adopted total correlation this time, there can be some other construction methods which show higher performance.

Still, we believe that this is one of the best acquisition performances achieved ever and will be an important step to truly practical lexical knowledge acquisition. Setting our future direction on the completely automatic construction of reliable thesaurus or ontology, the approach proposed here is to be applied to and integrated with various kinds of lexical knowledge acquisition methods in the future.

Acknowledgments

The author would like to thank Assoc. Prof. Katsuhiko Toyama and Assis. Prof. Yasuhiro Ogawa for their kind supervision and advice.

References

Ted Briscoe, John Carroll and Rebecca Watson. 2006. The Second Release of the RASP System. *Proc. of the COLING/ACL 06 Interactive Presentation Sessions*, 77–80.

Collins. 2002. Collins Cobuild Major New Edition CD-ROM. HarperCollins Publishers.

James R. Curran and Marc Moens. 2002. Improvements in automatic thesaurus extraction. In Workshop on Unsupervised Lexical Acquisition. *Proc. of ACL SIGLEX*, 231–238.

Christiane Fellbaum. 1998. *WordNet: an electronic lexical database*, MIT Press.

Maayan Geffet and Ido Dagan. 2004. Feature Vector Quality and Distributional Similarity. *Proc. of COLING 04*, 247–253.

Zellig Harris. 1985. Distributional Structure. Jerrold J. Katz (ed.) *The Philosophy of Linguistics*. Oxford University Press, 26–47.

Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. *Proc. of COLING 92*, 539–545.

Donald Hindle. 1990. Noun classification from predicate-argument structures. *Proc. of ACL 90*, 268–275.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. *Proc. of COLING/ACL 98*, 786–774.

Shachar Mirkin, Ido Dagan, and Maayan Geffet. 2006. Integrating pattern-based and distributional similarity methods for lexical entailment acquisition. *Proc. of COLING/ACL 06*, 579–586.

Brian Roark and Eugene Charniak. 1998. Noun phrase cooccurrence statistics for semi-automatic lexicon construction. *Proc. of COLING/ACL 98*, 1110–1116.

Roget. 1995. *Roget's II: The New Thesaurus, 3rd ed.* Houghton Mifflin.

Gerda Ruge. 1997. Automatic detection of thesaurus relations for information retrieval applications. *Foundations of Computer Science: Potential - Theory - Cognition*, LNCS, Volume 1337, 499–506, Springer Verlag.

Rion Snow, Daniel Jurafsly, and Andrew Y. Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems (NIPS) 17*.

Julie Weeds, David Weir and Diana McCarthy. 2004. Characterising Measures of Lexical Distributional Similarity. *Proc. of COLING 04*, 1015–1021.

An Integrated Architecture for Generating Parenthetical Constructions

Eva Banik
Department of Computing
The Open University
Walton Hall, Milton Keynes,
e.banik@open.ac.uk

Abstract

The aim of this research is to provide a principled account of the generation of embedded constructions (called *parentheticals*) and to implement the results in a natural language generation system. Parenthetical constructions are frequently used in texts written in a good writing style and have an important role in text understanding. We propose a framework to model the rhetorical properties of parentheticals based on a corpus study and develop a unified natural language generation architecture which integrates syntax, semantics, rhetorical and document structure into a complex representation, which can be easily extended to handle parentheticals.

1 Introduction

Parentheticals are constructions that typically occur embedded in the middle of a clause. They are not part of the main predicate-argument structure of the sentence and are marked by special punctuation (e.g. parentheses, dashes, commas) in written texts, or by special intonation in speech.

Syntactically, parentheticals can be realized by many different constructions, e.g.: appositive relative clauses (1a), non-restrictive relative clauses (1b), participial clauses (1c) or subordinate clauses (1d).

(1) a The new goal of the Voting Rights Act [– more minorities in political office –] is laudable. (wsj1137)

b GE, [which vehemently denies the government's allegations,] denounced Mr. Greenfield's suit. (wsj0617)

c But most businesses in the Bay area, [including Silicon Valley,] weren't greatly affected. (wsj1930)

d So far, [instead of teaming up,] GE Capital staffers and Kidder investment bankers have bickered. (wsj0604)

A common characteristics of parentheticals is that they express information that is not central to the meaning of the overall message conveyed by a text or spoken utterance and since they are specifically marked by punctuation or intonation, they allow the reader to distinguish between more and less important parts of the message. By structuring information this way, parentheticals make it easier for readers to decode the message conveyed by a text. Consider for example the following message that has been expressed by two different texts: one without parentheticals (2a) and one that contains two parentheticals (2b).

(2) a Eprex is used by dialysis patients who are anaemic. Prepulsid is a gastro-intestinal drug. Eprex and Prepulsid did well overseas.

b Eprex, [used by dialysis patients who are anaemic,] and Prepulsid, [a gastro-intestinal drug,] did well overseas. (wsj1156)

Parentheticals have been much studied in linguistics (see (Dehe and Kavalova, 2007), (Burton-Roberts, 2005) for a recent overview) but so far they

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 7–12,
Columbus, June 2008.

have received less attention in computational linguistics. Only a few studies have attempted a computational analysis of parentheticals, the most recent ones being (Bonami and Godard, 2007) who give an underspecified semantics account of evaluative adverbs in French and (Siddharthan, 2002) who develops a statistical tool for summarisation that separates parentheticals from the sentence they are embedded in. Both of these studies are limited in their scope as they focus on a very specific type of parentheticals.

From the perspective of natural language generation (NLG), as far as we know, nobody has attempted to give a principled account of parentheticals, even though these constructions contribute to the easy readability of generated texts, and therefore could significantly enhance the performance of NLG systems (Scott and Souza, 1990).

Most existing natural language generation systems use rhetorical structure to construct a text plan and map arguments of rhetorical relations onto individual sentences or clauses. As a result, the arguments of the same rhetorical relation will always occur immediately next to each other, although the surface realization of individual arguments may vary and a clause may appear *syntactically* embedded within the preceding clause. This linear succession of rhetorical relations and their arguments makes the generated text appear monotonous and staccato. As commonly mentioned by style manuals,[1] using different kinds of clause-combining strategies (e.g. semicolons, dash-interpolations, appositives) shows a clearer writing style.

The goal of this research is to give a principled account of parenthetical constructions and incorporate its findings into a natural language generation system.

2 System Architecture

We propose an integrated generation architecture for this purpose which uses a Tree Adjoining Grammar (Joshi, 1987) to represent linguistic information at all levels, including syntax, rhetorical structure and document structure.

Our approach is to make the elementary trees in the grammar as complex as possible, so that constraints on which trees can be combined with each other will be localized in the trees themselves. By incorporating information about rhetorical structure and document structure into the trees, we are extending the domain of locality of elementary trees as much as possible and this allows the generator to keep the global operations for combining trees as simple as possible. This approach has been referred to as the 'Complicate Locally, Simplify Globally' principle (Joshi, 2004).

The input to the generator is a set of rhetorical relations and semantic formulas. For each formula the system selects a set of trees from the grammar, resulting in a number of possible tree sets associated with the input.

The next step is to filter out sets of trees that will not lead to a possible realization. In the current implementation this is achieved by a version of polarity filtering where we associate not only the syntactic categories of root, substitution and foot nodes with a positive or negative value (Gardent and Kow, 2006) but also add the semantic variable associated with these nodes. The values summed up by polarity filtering are [node, semantic variable] pairs, which represent restrictions on possible syntactic realizations of semantic (or rhetorical) arguments.

Parentheticals often pose a problem for polarity filtering because in many cases there is a shared element between the parenthetical and its host, which normally occurs twice in non-parenthetical realizations of the same input, but only once when there is a parenthetical. (e.g., in (2a) the NP 'Eprex' occurs twice, but only once in (2b)). In order to allow for this variation, when summing up the values for substitution and root nodes we consider multiple occurrences of NP substitution nodes associated with the same semantic variable as if they were a single instance. This results in one or more NP substitution nodes left empty at the end of the derivation, which are then filled with a pronoun by a referring expression module at the final stage of the generation process.

3 Corpus Study

The generator is informed by a corpus study of embedded discourse units on two discourse annotated corpora: the RST Discourse Treebank (Carlson et al., 2001) and the Penn Discourse Treebank (PDTB-

[1] See for example, Rule 14 of (Strunk and White, 1979)

		Elab-add	Example	Elab-gen-spec	Restatement	Elab-set-mem	Attribution	Condition	Antithesis	Concession	Circumstance	Purpose	
NP-modifiers	relative clause	**143**		2		2							147
	participial clause	**96**	4			1	1				11	4	117
	NP	**34**		**8**	**22**								64
	NP-coord					6							6
	cue + NP	5	1						2	3	2		13
	Adj + cue	2											2
	number	2											2
	including + NP		**13**			5							18
VP- or S- modifiers	to-infinitive	4										**30**	34
	NP + V						**106**						106
	cue + S	5						**20**	**14**	**9**	**29**		77
	PP	11									9	1	21
	S	7	1	1									9
	according to NP						7						7
	V + NP						6						6
	as + S						4						4
	Adv + number	1									1		2
	cue + Adj										2		2
	cue + participial								2				2
	cue + V						1						1
		310	19	11	22	14	125	20	18	12	54	35	640

Table 1: Syntactic types of parentheticals in the RST corpus

Relation	Connective in parenthetical	Connective in host	distribution in corpus
TEMPORAL	101 (48.8%)	2	3434 (18.6%)
CONTINGENCY	53 (25.6%)	0	3286 (17.8%)
COMPARISON	38 (18.3%)	5	5490 (29.7%)
EXPANSION	15 (7.2%)	5	6239 (33.8%)
TOTAL:	207	12	18484

Table 2: Relations between parentheticals and their hosts in the PDTB

Group, 2008).[2] The aim of the study was to establish what rhetorical relations can hold between parentheticals and their hosts and whether individual rhetorical relations tend to correlate with specific syntactic types.

Table 1 illustrates the findings of the study on the RST corpus, showing the correlation between syntactic types of parentheticals and rhetorical relations between parentheticals and their hosts in the corpus. The majority of parentheticals in this study were syntactically related to their hosts and they can be divided into two main groups. The most frequently occurring type is ELABORATION/EXPANSION-type NP-modifiers which are realized by relative clauses, NPs or nominal postmodifiers with non-finite clauses and express some type of ELABORATION, EXAMPLE or RESTATEMENT relation. 73.4% of parentheticals belong to this group in the RST corpus.

The other type of parentheticals are NON-ELABORATION/EXPANSION-type VP- or S-modifiers, which are realized by subordinate clauses, to-infinitives and PPs and express CIRCUMSTANCE, PURPOSE, CONDITION, ANTITHESIS,or CONCESSION relations. 26.6% of parentheticals in the corpus belong to this group.

Because of the decision taken in the PDTB to only annotate clausal arguments of discourse connectives, parentheticals found in this corpus are almost

[2]The details of this study are reported in (Banik and Lee, 2008)

all subordinate clauses, which is clearly an artifact of the annotation guidelines. This corpus only annotates parentheticals that contain a discourse connective and we have found that in almost all cases the connective occurs within the parenthetical. We have found only 12 discourse adverbs that occurred in the host sentence.

The present corpus study is missing several types of parentheticals because of the nature of the annotation guidelines of the corpora used. For example, in the RST corpus some phrasal elements that contain a discourse connective (3a) and adjectives or reduced relative clauses that contain an adjective without a verbal element are not annotated (3b):

(3) a But the technology, [while reliable,] is far slower than the widely used hard drives. (wsj1971)

b Each $5000 bond carries one warrant, [exercisable from Nov. 28, 1989, through Oct. 26, 1994] to buy shares at an expected premium of 2 1/2 % to the closing share price when terms are fixed Oct. 26. (wsj1161)

These constructions are clear examples of parentheticals and we would expect them to behave similarly to subordinating conjunctions and relative clauses respectively. As a test case we decided to allow adjectives to function as parentheticals in the grammar of the generator and if the results are evaluated as satisfactory, plan to extend this analysis to other constructions not covered by our corpus study.

4 Generating Parentheticals — An Example

We associate auxiliary trees with parenthetical occurrences of the most frequently embedded rhetorical relations based on the above corpus study.

The basic assumption behind assigning syntactic trees to parenthetical rhetorical relations is that the semantic type of the arguments of the relation should be mirrored by their syntax. Thus if one of the arguments of a rhetorical relation is an object then it must be represented by an NP in the syntax; if it is a proposition then it must be assigned an S- or VP-auxiliary tree. The satellite of the rhetorical relation is always substituted into the auxiliary tree,

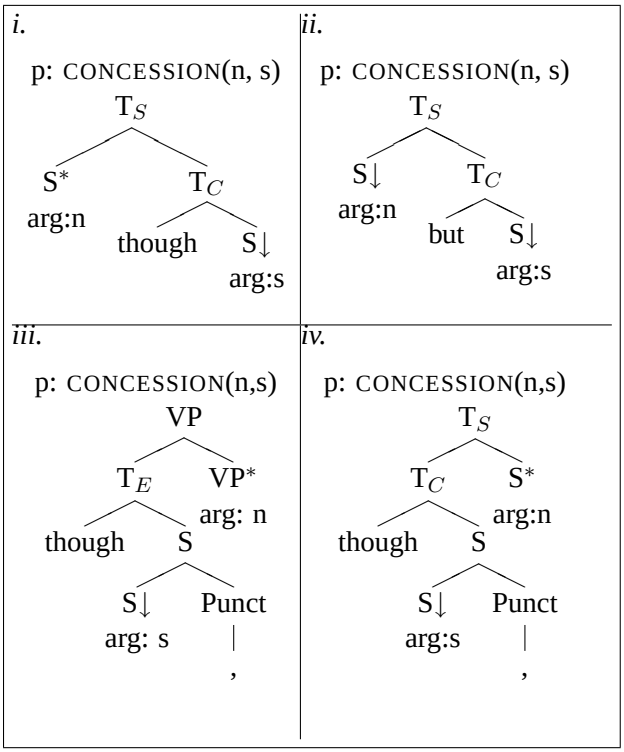


Figure 1: Elementary trees for CONCESSION

and the nucleus is associated with the footnode (this later gets unified with the semantic label of the tree that the auxiliary tree adjoins to).

Figure 1 illustrates four elementary trees for the CONCESSION relation. The trees in boxes *i.* and *ii.* correspond to regular uses of CONCESSION while the trees in *iii.* and *iv.* correspond to its parenthetical occurrences. Using these trees along with the elementary trees in Figure 3, and given the input below, the system generates the following five possible realizations:

```
   Input:  [[l3, concession, l1, l2], [l1,legal,x], [l2, fatal,
x], [x,substance]]

   Output:

1.  the substance, though it is fatal, is legal

2.  the substance is legal though it is fatal

3.  though it is fatal, the substance is legal

4.  though the substance is fatal, it is legal

5.  the substance is legal but it is fatal
```

Figure 2 gives the elementary trees assigned to

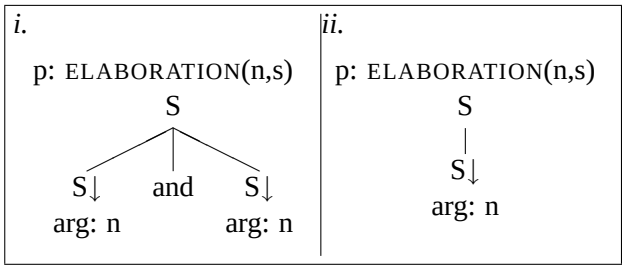


Figure 2: Elementary trees for ELABORATION

the most frequently occurring parenthetical rhetorical relation, ELABORATION-ADDITIONAL. The tree in box *i.* is associated with non-parenthetical uses of the relation, and box *ii.* shows the tree used for parenthetical ELABORATION. Since in parenthetical uses of ELABORATION the two arguments of the relation combine with each other and not with a third tree, as in the case of parenthetical CONCESSION, the role of the lexically empty parenthetical tree in box *ii.* is to restrict the type of tree selected for the nucleus of ELABORATION. Since the satellite has to end up as the parenthetical, the nucleus has to be restricted to the main clause, which is achieved by associating its semantic variable with an S substitution node in the tree.

To give an example, Figure 3. illustrates elementary trees for the input below:

```
   Input:  [[l3, elaboration, l1, l2], [l1,illegal,x], [l2,
fatal, x], [x,substance]]

   Output:

1.  the fatal substance is illegal

2.  the substance, which is fatal, is illegal

3.  the substance is illegal and it is fatal
```

The parenthetical ELABORATION tree is used for constructing outputs 1. and 2., which restricts the nucleus to select the initial tree in box *iii.* on Figure 3. As a result, the satellite of the relation has to select on of the auxiliary trees in box *i.* or *ii.* in order to be able to combine with the nucleus. The case where both satellite and nucleus are assigned initial trees is handled by the non-parenthetical tree in box *i.* on Figure 2.

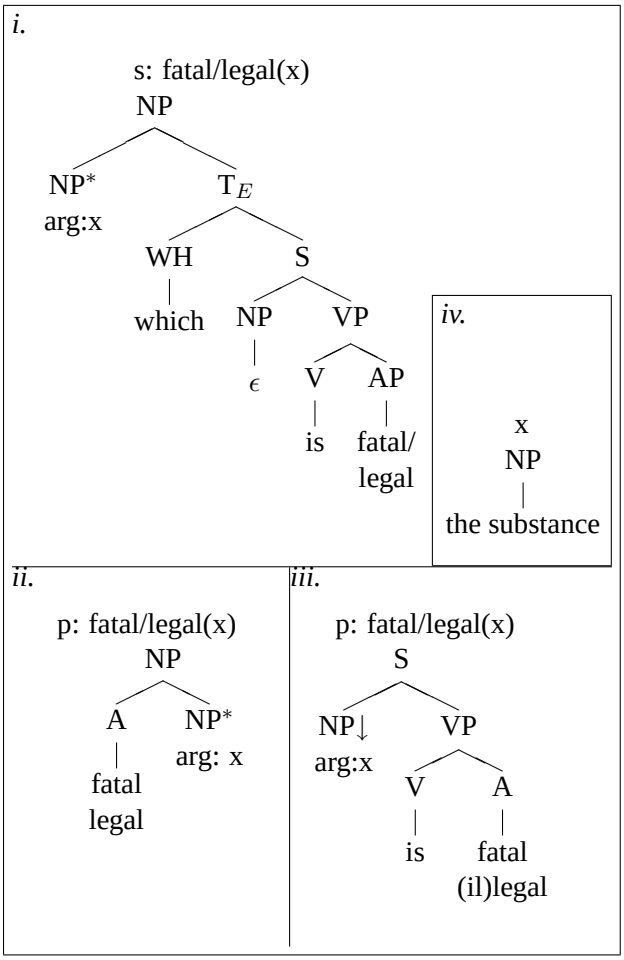


Figure 3: Elementary TAG trees for semantic formulas

5 Directions for further research

A possible way to control the generator is to enrich the input representation by adding restrictions on the types of trees that are allowed to be selected, similarly to (Gardent and Kow, 2007) (e.g., if a rhetorical relation is restricted to selecting initial trees for its satellite then it won't be generated as a parenthetical). Another way to select a single output is to establish ranking constraints (these could depend, e.g., on the genre of the text to be generated) and choose the top ranked candidate for output.

At the moment the elementary trees in the grammar contain document structure nodes (Power et al., 2003) which are not used by the generator. We plan to extend the analysis of parentheticals to big-

ger structures like footnotes or a paragraph separated in a box from the rest of the text and the document structure nodes in the elementary trees will be used to generate these.

Given the small size of the grammar, currently polarity filtering is enough to filter out just the grammatical realizations from the set of possible treesets. As the grammar size increases we expect that we will need additional constraints to reduce the number of possible tree sets selected for a given input. Also, once the generator will be capable of handling longer inputs, we will need to avoid generating too many parentheticals. Both the number of possible tree sets and the number of parentheticals in the outputs could be reduced by allowing the generator to select parenthetical realizations for only a predefined percentage of each rhetorical relation in the input. This number can be first obtained from our corpus study, and fine-tuned based on evaluations of the generated output.

The current implementation uses a very simplistic referring expression module which inserts a pronoun in every NP position left open at the end of the derivation, unless it is in a sentence initial position. Parentheticals often involve the use of referring expressions and can sound more natural when the embedded constituent involves a reference to an element in the main clause, therefore a more sophisticated algorithm for referring expression generation will be used in the future.

Although our corpus study gives important information about which rhetorical relation to realize as a parenthetical, how often, and using which syntactic construction, there seem to be additional restrictions on the use of certain parentheticals. Consider for example the two realizations (4 a and b) of the CONCESSION relation below where the parenthetical in (4b) sounds very unnatural:

```
   concession:
n:  a few people may experience side-effects
s:  most people benefit from taking Elixir
```

(4) a Though most people benefit from taking Elixir, a few people may experience side-effects.

b ?? A few people, though most people benefit from taking Elixir, may experience side-effects.

References

E. Banik and A. Lee. 2008. A study of parentheticals in discourse corpora – implications for NLG systems. In *Proceedings of LREC 2008, Marrakesh*.

O. Bonami and D. Godard. 2007. Parentheticals in underspecified semantics: The case of evaluative adverbs. *Research on Language and Computation*, 5(4):391–413.

N. Burton-Roberts. 2005. Parentheticals. In E. K. Brown, editor, *Encyclopaedia of Language and Linguistics*. Elsevier Science, 2nd edition edition.

L. Carlson, D. Marcu, and M. E. Okurowski. 2001. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Proceedings of the Second SIGdial Workshop on Discourse and Dialogue*, pages 1–10, Morristown, NJ, USA. Association for Computational Linguistics.

N. Dehe and Y. Kavalova, editors, 2007. *Parentheticals*, chapter Parentheticals: An introduction, pages 1–22. Linguistik aktuell Linguistics today 106. Amsterdam Philadelphia: John Benjamins.

C. Gardent and E. Kow. 2006. Three reasons to adopt tag-based surface realisation. In *The Eighth International Workshop on Tree Adjoining Grammar and Related Formalisms (TAG+8)*, Sydney/Australia.

C. Gardent and E. Kow. 2007. A symbolic approach to near-deterministic surface realisation using tree adjoining grammar. In *In 45th Annual Meeting of the ACL*.

A. K. Joshi. 1987. The relevance of tree adjoining grammar to generation. In G. Kempen, editor, *Natural Language Generation*, pages 233–252. Martinus Nijhoff Press, Dordrect, The Netherlands.

A. K. Joshi. 2004. Starting with complex primitives pays off: complicate locally, simplify globally. *Cognitive Science: A Multidisciplinary Journal*, 28(5):637–668.

PDTB-Group. 2008. The Penn Discourse Treebank 2.0 Annotation Manual. Technical Report IRCS-08-01, Institute for Research in Cognitive Science, University of Pennsylvania.

R. Power, D. Scott, and N. Bouayad-Agha. 2003. Document structure. *Computational Linguistics*, 29(4):211–260.

D. Scott and C. S. Souza. 1990. Getting the message across in RST-based text generation. In C. Mellish R. Dale M. Zock, editor, *Current Research in Natural Language Generation*, pages 31–56. Academic Press.

A. Siddharthan. 2002. Resolving attachment and clause boundary ambiguities for simplifying relative clause constructs. In *Student Research Workshop, ACL*.

W. Jr. Strunk and E. B. White. 1979. *The Elements of Style*. Macmillan, third edition.

Inferring Activity Time in News through Event Modeling

Vladimir Eidelman
Department of Computer Science
Columbia University
New York, NY 10027
vae2101@columbia.edu

Abstract

Many applications in NLP, such as question-answering and summarization, either require or would greatly benefit from the knowledge of when an event occurred. Creating an effective algorithm for identifying the activity time of an event in news is difficult in part because of the sparsity of explicit temporal expressions. This paper describes a domain-independent machine-learning based approach to assign activity times to events in news. We demonstrate that by applying topic models to text, we are able to cluster sentences that describe the same event, and utilize the temporal information within these event clusters to infer activity times for all sentences. Experimental evidence suggests that this is a promising approach, given evaluations performed on three distinct news article sets against the baseline of assigning the publication date. Our approach achieves 90%, 88.7%, and 68.7% accuracy, respectively, outperforming the baseline twice.

1 Introduction

Many practical applications in NLP either require or would greatly benefit from the use of temporal information. For instance, question-answering and summarization systems demand accurate processing of temporal information in order to be useful for answering 'when' questions and creating coherent summaries by temporally ordering information. Proper processing is especially relevant in news, where multiple disparate events may be described within one news article, and it is necessary to identify the separate timepoints of each event.

Event descriptions may be confined to one sentence, which we establish as our text unit, or be spread over many, thus forcing us to assign all sentences an activity time. However, only 20%-30% of sentences contain an explicit temporal expression, thus leaving the vast majority of sentences without temporal information. A similar proportion is reported in Mani et al. (2003), with only 25% of clauses containing explicit temporal expressions. The sparsity of these expressions poses a real challenge. Therefore, a method for efficiently and accurately utilizing temporal expressions to infer activity times for the remaining 70%-80% of sentences with no temporal information is necessary.

This paper proposes a domain-independent machine-learning based approach to assign activity times to events in news without deferring to the publication date. Posing the problem in an information retrieval framework, we model events by applying topic models to news, providing a way to automatically distribute temporal information to all sentences. The result is prototype system which achieves promising results.

In the following section, we discuss related work in temporal information processing. Next we motivate the use of topic models for our task, and present our methods for distributing temporal information. We conclude by presenting and discussing our results.

2 Related Work

Mani and Wilson (2000) worked on news and introduced an annotation scheme for temporal expressions, and a method for using explicit tempo-

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 13–18,
Columbus, June 2008.

Sentence Order	Event	Temporal Expression
1	Event X	None
2	Event Y	January 10, 2007
3	Event X	None
4	Event X	November 16, 1967
5	Event Y	None
6	Event Y	January 10, 2007
7	Event X	None

Table 1: Problematic Example

ral expressions to assign activity times to the entirety of an article. Their preliminary work on inferring activity times suggested a baseline method which spread time values of temporal expressions to neighboring events based on proximity. Filatova and Hovy (2001) also process explicit temporal expressions within a text and apply this information throughout the whole article, assigning activity times to all clauses.

More recent work has tried to temporally anchor and order events in news by looking at clauses (Mani et al., 2003). Due to the sparsity of temporal expressions, they computed a reference time for each clause. The reference time is inferred using a number of linguistic features if no explicit reference is present, but the algorithm defaults to assigning the most recent time when all else fails.

A severe limitation of previous work is the dependence on article structure. Mani and Wilson (2000) attribute over half the errors of their baseline method to propagation of an incorrect event time to neighboring events. Filatova and Hovy (2001) infer time values based on the most recently assigned date or the date of the article. The previous approaches will all perform unfavorably in the example presented in Table 1, where a second historical event is referred to between references to a current event. This kind of example is quite common.

3 Modeling News

To address the aforementioned issues of sparsity while relieving dependence on article structure, we treat event discovery as a clustering problem. Clustering methods have previously been used for event identification (Hatzivassiloglou et al., 2000; Siddharthan et al., 2004). After a topic model of news text is created, sentences are clustered into topics - where each topic represents a specific event. This allows us to utilize all available temporal information in each cluster to distribute to all the sentences within that cluster, thus allowing for assigning of activity times to sentences without explicit temporal expressions. Our key assumption is that similar sentences describe the same event.

Our approach is based on information retrieval techniques, so we subsequently use the standard language of text collections. We may refer to sentences, or clusters of sentences created from a topic model as 'documents', and a collection of sentences, or collection of clusters of sentences from one or more news articles as a 'corpus'. We use Latent Dirichlet Allocation (LDA) (Blei et al., 2003), a generative model for describing collections of text corpora, which represents each document as a mixture over a set of topics, where each topic has associated with it a distribution over words. Topics are shared by all documents in the corpus, but the topic distribution is assumed to come from a Dirichlet distribution. LDA allows documents to be composed of multiple topics with varying proportions, thus capturing multiple latent patterns.

Depending on the words present in each document, we associate it with one of N topics, where N is the number of latent topics in the model. We assign each document to the topic which has the highest probability of having generated that document. We expect document similarity in a cluster to be fairly high, as evidenced by document modeling performance in Blei et al. (2003). Since each cluster is a collection of similar documents, with our assumption that similar documents describe the same event, we conclude that each cluster represents a specific event. Thus, if at least one sentence in an event cluster contains an explicit temporal expression, we can distribute that activity time to other sentences in the cluster using an inference algorithm we explain in the next section. More than one event cluster may represent the same event, as in Table 3, where both topics describe a different perspective on the same event: the administrative reaction to the incident at Duke.

Creating a cluster of similar documents which represent an event can be powerful. First, we are no longer restricted by article structure. To refer back to

Table 1, our approach will assign the correct activity time for all event X sentences, even though they are separated in the article and only one contains an explicit temporal expression, by utilizing an event cluster which contains the four sentences describing event X to distribute the temporal information[1].

Second, we are not restricted to using only one article to assign activity times to sentences. In fact, one of the major strengths of this approach is the ability to take a collection of articles and treat them all as one corpus, allowing the model to use all explicit temporal expressions on event X present throughout all of the articles to distribute activity times. This is especially helpful in multidocument summarization, where we have multiple articles on the same event.

Additionally, using LDA as a method for event identification may be advantageous over other clustering methods. For one, Siddharthan et al. (2004) reported that removing relative clauses and appositives, which provide background or discourse related information, improves clustering. LDA allows us to discover the presence of multiple events within a sentence, and future work will focus on exploiting this to improve clustering.

3.1 Corpus

We obtained 22 news articles, which can be divided into three distinct sets: Duke Rape Case (DR), Terrorist Bombings in Mumbai (MB), Israeli-Lebanese conflict (IC) (Table 2). All articles come from English Newswire text, and each sentence was manually annotated with an activity time by people outside of the project. The Mumbai Bombing articles all occur within a several day span, as do the Israeli-Conflict articles. The Duke Rape case articles are an exception, since they are comprised of multiple events which happened over the course of several months: Thus these articles contain many cases such as *"The report said...on March 14..."*, where the report is actually in May, yet speaks of events in March. For the purposes of this experiment we took the union of the possible dates mentioned in a sentence as acceptable activity times, thus both the report statement date and the date mentioned in the report are correct activity times for the sentence. Future work will investigate whether we can discriminate between these two dates.

[1] Analogously, our approach will assign correct activity time to all event Y sentences

Article Set	# of Articles	# of Sentences
Duke Rape Case	5	151
Mumbai Bombing	8	284
Israeli Conflict	9	300

Table 2: Article and Sentence distribution

Our approach relies on prior automatic linguistic processing of the articles by the Proteus system (Grishman et al., 2005). The articles are annotated with time expression tags, which assign values to both absolute *"July 16, 2006"* and relative *"now"* temporal expressions. Although present, our approach does not currently use activity time ranges, such as "past 2 weeks" or "recent days". The articles are also given entity identification tags, which assigns a unique intra-article id to entities of the types specified in the ACE 2005 evaluation. For example, both "they" - an anaphoric reference - and "police officers" are recognized as referring to the same real-world entity.

3.2 Feature Extraction

From this point on unless otherwise noted, reference to news articles indicates one of the three sets of news articles, not the complete set. We begin by breaking news articles into their constituent sentences, which are our 'documents', the collection of them being our 'corpus', and indexing the documents.

We use the bag-of-words assumption to represent each document as an unordered collection of words. This allows the representation of each document as a word vector. Additionally, we add any entity identification information and explicit temporal expressions present in the document to the feature vector representation of each document.

3.3 Intra-Article Event Representation

To represent events within one news article, we construct a topic model for each article separately. The Intra-Article (IAA) model constructed for an article allows us to group sentences within that article together according to event. This allows the formation of new 'documents', which consist not of single

The administrators did not know of the racial dimension until March 24, the report said. The report did say that Brodhead was hampered by the administration's lack of diversity. He said administrators would be reviewed on their performance on the normal schedule and he had no immediate plans to make personnel changes. Administrators allowed the team to keep practicing; Athletics Director Joe Alleva called the players "wonderful young men."
Yet even Duke faculty members, many of them from the '60s and '70s generations that pushed college administrators to ease their controlling ways, now are urging the university to require greater social as well as scholastic discipline from students. Duke professors, in fact, are offering to help draft new behavior codes for the school. With years of experience and academic success to their credit, faculty members ought to be listened to. For the moment, five study committees appointed by Brodhead seem to mean business, which is encouraging.

Table 3: Two topics representing a different perspective on the same event

sentences, but a cluster of sentences representing an event. Accordingly, we combine the feature vector representations of the single sentences in an event cluster into one feature vector, forming an aggregate of all their features. Although at this stage we have everything we need to infer activity times, our approach allows incorporating information from multiple articles.

3.4 Inter-Article Event Representation

To represent events over multiple articles, we suggest two methods for Inter-Article (IRA) topic modeling. The first, IRA.1, is to combine the articles and treat them as one large article. This allows processing as described in IAA, with the exception that event clusters may contain sentences from multiple articles. The second, IRA.2, builds on IAA models of single articles and uses them to construct an IRA model. The IRA.2 model is constructed over a corpus of documents containing event clusters, allowing a grouping of event clusters from multiple articles. Event clusters may now be composed of sentences describing the same event from multiple articles, thus increasing our pool of explicit temporal expressions available for inference.

3.5 Activity Time Assignment

To accurately infer activity times of all sentences, it is crucial to properly utilize the available temporal expressions in the event clusters formed in the IRA or IAA models. Our proposed inference algorithm is a starting point for further work. We use the most frequent activity time present in an event cluster as the value to assign all the sentences in that event cluster. In phase one of the algorithm we process each event cluster separately. If the majority of sentences with temporal expressions have the same activity time, then this activity time is distributed to the other sentences. If there is a tie between the number of occurrences of two activity times, both these times are distributed as the activity time to the other sentences. If there is no majority time and no tie in the event cluster, then each of the sentences with a temporal expression retains its activity time, but no information is distributed to the other sentences. Phase two of the inference algorithm reassembles the sentences back into their original articles, with most sentences now having activity times tags assigned from phase one. Sentences that remain unmarked, indicating that they were in event clusters with no majority and no tie, are assigned the majority activity time appearing in their reassembled article.

4 Empirical Evaluation

In evaluating our approach, we wanted to compare different methods of modeling events prior to performing inference.

- Method (1) IAA then IRA.2 - Creating IAA models with 20 topics for each news article, and IRA.2 models for each of the three sets of IAA models with 20, 50, and 100 topic.
- Method (2) IAA only - Creating an IAA model with 20 topics for each article
- Method (3) IRA.1 only - Creating IRA.1 model with 20 and 50 topics for each of the three sets of articles.

4.1 Results

Table 4 presents results for the three sets of articles on the six different experiments performed. Since our approach assigns activity times to all sentences, overall accuracy is measured as the total number of correct activity time assignments made out of the total number of sentences. The baseline accuracy is computed by assigning each sentence the article publication date, and because news generally describes current events, this achieves remarkably high performance.

The overall accuracy measures performance of the complete inference algorithm, while the rest of the metrics measure the performance of phase one only, where we process each event cluster separately. Assessing the performance of phase one allows us to indirectly evaluate the event clusters which we create using LDA. M1 accuracy represents the number of sentences that were assigned the correct activity time in phase one out of the total number of activity time inferences made in phase one. Thus, this does not take into account any assignments made by phase two, and allows us to examine our assumptions about event representation expressed earlier. A large denominator in M1 indicates that many sentences were assigned in phase one, while a low one indicates the presence of event clusters which were unable to distribute temporal information.

M2 looks at how well the algorithm performs on the difficult cases where the activity time is not the same as the publication date. M3 looks at how well the algorithm performs on the majority of sentences which have no temporal expressions.

For the IC and DR sets, results show that Method (1), where IAA is performed prior to IRA.2 achieves the best performance, with accuracy of 88.7% and 90%, respectively, giving credence to the claim that representing events within an article before combining multiple articles improves inference.

The MB set somewhat counteracts this claim, as the best performance was achieved by Method (3), where IRA.1 is performed. This may be due to the fact that MB differs from DR and IC sets in that it contains several regurgitated news articles. Regurgitated news articles are comprised almost entirely of statements made at a previous time in other news articles. Method (3) combines similar sentences from all the articles right away, placing sentences from regurgitated articles in an event cluster with the original sentences. This allows our approach to outperform the baseline system by 4.3%, with and accuracy of 68.7%.

5 Discussion

There are limitations to our approach which need to be addressed. Foremost, evidence suggests that event clusters are not perfect, as error analysis has shown event clusters which represent two or more

Set	Setup	Accur.	M1	M2	M3
DR	Base	**135/151**			
		89.4%			
DR	(1) 20	121/151	55/83	5/12	27/43
		80.1%	66.2%	41.6%	62.7%
DR	(1) 50	**136/151**	91/105	4/13	60/66
		90.0%	86.6%	30.7%	90.9%
DR	(1)100	128/151	87/109	4/13	58/70
		84.7%	79.8%	30.7%	82.8%
DR	(2) 20	106/151	45/68	4/11	20/33
		70.2%	66.2%	36.4%	60.6%
DR	(3) 20	111/151	82/110	8/14	49/71
		73.5%	74.7%	57.1%	69.0%
DR	(3) 50	99/151	92/135	6/14	63/95
		65.5%	68.1%	42.9%	66.3%
Set	Setup	Accur.	M1	M2	M3
MB	Base	**183/284**			
		64.4%			
MB	(1) 20	166/284	116/187	41/68	60/104
		58.5%	62.0%	60.2%	57.7%
MB	(1) 50	152/284	121/206	41/72	66/120
		53.5%	58.7%	56.9%	55.0%
MB	(1)100	139/284	112/204	41/81	60/124
		48.9%	54.9%	50.6%	48.4%
MB	(2) 20	143/284	103/161	40/63	49/85
		50.3%	63.9%	63.5%	57.3%
MB	(3) 20	146/284	99/160	45/64	47/81
		51.4%	61.9%	70.3%	58.0%
MB	(3) 50	**195/284**	123/184	32/67	74/103
		68.7%	66.8%	47.8%	71.8%
Set	Setup	Accur.	M1	M2	M3
IC	Base	**272/300**			
		90.7%			
IC	(1) 20	250/300	158/205	12/22	118/151
		83.3%	77.1%	54.5%	78.1%
IC	(1) 50	263/300	168/192	12/19	127/139
		87.7%	87.5%	63.2%	91.4%
IC	(1)100	**266/300**	173/202	11/20	130/149
		88.7%	85.6%	55.0%	87.2%
IC	(2) 20	250/300	156/181	11/18	117/130
		83.3%	86.2%	61.1%	90.0%
IC	(3) 20	225/300	112/145	14/21	75/95
		75.0%	77.2%	66.7%	78.9%
IC	(3) 50	134/300	115/262	14/25	76/206
		44.7%	43.9%	56.0%	36.9%

Table 4: Results : Sentence Breakdown

events. Event clusters which contain sentences describing several events pose a real challenge, as they are primarily responsible for inhibiting performance. This limitation is not endemic to our approach for event discovery, as Xu et al. (2006) stated that event extraction is still considered as one of the most challenging tasks, because an event mention can be expressed by several sentences and different linguistic expressions.

One of the major strengths of our approach is the ability to combine all temporal information on an event from multiple articles. However, due the imperfect event clusters, combining temporal information from different articles within an event cluster has not yet yielded satisfactory results.

Although sentences from the same article in IRA event clusters usually represent the same event, other sentences from different articles may not. We modified the inference algorithm to reflect this, and only consider sentences from the same news article when distributing temporal information, even though sentences from other articles may be present in the event cluster. Therefore, further work to construct event clusters which more closely represent events is expected to yield improvements in performance. Future work will explore a richer feature set, including such features as cross-document entity identification information, linguistic features, and outside semantic knowledge to increase robustness of the feature vectors. Finally, the optimal model parameters are currently selected by an oracle, however, we hope to further evaluate our approach on a larger dataset in order to determine how to automatically select the optimal parameters.

6 Conclusion

This paper presented a novel approach for inferring activity times for all sentences in a text. We demonstrate we can produce reasonable event representations in an unsupervised fashion using LDA, posing event discovery as a clustering problem, and that event clusters can further be used to distribute temporal information to the sentences which lack explicit temporal expressions. Our approach achieves 90%, 88.7%, and 68.7% accuracy, outperforming the baseline set forth in two cases. Although differences prevent a direct comparison, Mani and Wilson (2000) achieved an accuracy of 59.4% on 694 verb occurrences using their baseline method, Filatova and Hovy (2001) achieved 82% accuracy on time-stamping clauses for a single type of event on 172 clauses, and Mani et al. (2003) achieved 59% accuracy in their algorithm for computing a reference time for 2069 clauses. Future work will improve upon the majority criteria used in the inference algorithm, on creating more accurate event representations, and on determining optimal model parameters automatically.

Acknowledgements

We wish to thank Kathleen McKeown and Barry Schiffman for invaluable discussions and comments.

References

David M. Blei, Andrew Y. Ng and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, vol. 3, pp.993–1022

Elena Filatova and Eduard Hovy. 2001. Assigning Time-Stamps to Event-Clauses. *Workshop on Temporal and Spatial Information Processing, ACL'2001* 88-95.

Ralph Grishman, David Westbrook, and Adam Meyers. 2005. NYU's English ACE 2005 system description. *In ACE 05 Evaluation Workshop*.

Vasileios Hatzivassiloglou, Luis Gravano, and Ankineedu Maganti. 2000. An Investigation of Linguistic Features and Clustering Algorithms for Topical Document Clustering. *In Proceedings of the 23rd ACM SIGIR*, pages 224-231.

Inderjeet Mani, Barry Schiffman and Jianping Zhang. 2003. Inferring Temporal Ordering of Events in News. *Proceedings of the Human Language Technology Conference*.

Inderjeet Mani and George Wilson. 2000. Robust Temporal Processing of News. *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, 69-76. Hong Kong.

Advaith Siddharthan, Ani Nenkova, and Kathleen McKeown. 2004. Syntactic simplification for improving content selection in multi-document summarization. *In 20th International Conference on Computational Linguistics* .

Feiyu Xu, Hans Uszkoreit, and Hong Li. 2006. Automatic event and relation detection with seeds of varying complexity. *In Proceedings of the AAAI Workshop Event Extraction and Synthesis*, pages 1217, Boston.

Combining Source and Target Language Information for Name Tagging of Machine Translation Output

Shasha Liao
New York University
715 Broadway, 7th floor
New York, NY 10003 USA
liaoss@cs.nyu.edu

Abstract

A Named Entity Recognizer (NER) generally has worse performance on machine translated text, because of the poor syntax of the MT output and other errors in the translation. As some tagging distinctions are clearer in the source, and some in the target, we tried to integrate the tag information from both source and target to improve target language tagging performance, especially recall.

In our experiments with Chinese-to-English MT output, we first used a simple merge of the outputs from an ET (Entity Translation) system and an English NER system, getting an absolute gain of 7.15% in F-measure, from 73.53% to 80.68%. We then trained an MEMM module to integrate them more discriminatively, and got a further average gain of 2.74% in F-measure, from 80.68% to 83.42%.

1 Introduction

Because of the growing multilingual environment for NLP, there is an increasing need to be able to annotate and analyze the output of machine translation (MT) systems. But treating this task as one of processing "ordinary text" can lead to poor results. We examine this problem with respect to the name tagging of English text.

A Named Entity Recognizer (NER) trained on an English corpus does not have the same performance when applied to machine-translated text. From our experiments on NIST 05 Chinese-to-English MT evaluation data, when we used the same English NER to tag the reference translation and the MT output, the F-measure was 81.38% for the reference but only 73.53% for the MT output. There are two primary reasons for this. First, the performance of current translation systems is not very good, and so the output is quite different from Standard English text. The fluency of the translated text will be poor, and the context of a named entity may be weird. Second, the translated text has some foreign names which are hard for the English NER to recognize, even if they are well translated by the MT system, because such names appear very infrequently in the English training corpus.

Training an NER on MT output does not seem to be an attractive solution. It may take a lot of time to manually annotate a large amount of training data, and this labor may have to be repeated for a new MT system or even a new version of an existing MT system. Furthermore, the resulting system may still not work well, in so far as the translation is not good and information is somehow distorted. In fact, sometimes the meanings of the translated sentences are hard to decipher unless we check the source language or get a human translated document as reference. As a result, we need source language information to aid the English NER.

However, it is also not enough to rely entirely on the source language NE results and map them onto the translated English text. First, the word alignment from source language to English generated by the MT system may not be accurate, leading to problems in mapping the Chinese name tags. Second, the translated text is not exactly same as the source language because there may be information missed or added. For example, the Chinese phrase "*香港地铁*", which is not a name in Chinese, and should be literally translated as

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 19–24,
Columbus, June 2008.

"*the subway in Hong Kong*", may end up being translated to "*mtrc*", the abbreviation of "*The Mass Transit Railway Corporation*", which is an organization in Hong Kong (and so should get a name tag in English).

If we can use the information from both the source language and the translated text, we cannot only find the named entities missed by the English NER, but also modify incorrect boundaries in the English results which are caused by the bad content. However, using word alignment to map the source language information into the English text is problematic, for two reasons: First, the word alignment produced by machine translation is typically not very good, with a Chinese-English AER (alignment error rate) of about 40% (Deng and William 2005). So just using word alignment to map the information would introduce a lot of noise. Second, in the case of function words in English which have no corresponding realization in Chinese, traditional word alignment would align the function word with another Chinese constituent, such as a name, which could lead to boundary errors in tagging English names. We have therefore used an alternative method to fetch the source language information for information extraction, which is called Entity Translation and is described in Section 3.

2 Motivation

When we use the English NER to annotate the translated text, we find that the performance is not as good as English texts. This is due to several types of problems.

2.1 Bad name contexts

Producing correct word order is very hard for a phrase-based MT system, particularly when translating between two such disparate languages, and there are still a lot of Chinese syntax structures left in translated text, which are usually not regular English expressions. As a result, it is hard for the English NER to detect names in these contexts.[1]

Ex. 1. annan said, "**kumaratunga** president personally against him to areas under guerrilla control field visit because it feared the rebels will use his visit as a political chip"

It is hard to recognize from this example that ***kumaratunga*** is a person name unless we are already familiar with this name or realize this is a normal Chinese expression structure, although not an English one.

Ex. 2. A reporter from **shantou** <ORG[2]> **university** school of medicine</ORG>, faculty of medicine, **university of** <GPE>**hong kong**</GPE>, <ORG>influenza research center</ORG> was informed that …...

Here source language information can help fix incorrect name boundaries assigned by the English NER, especially from a messy context. In Example 3, the source language tagger can tell us that "shantou university" and "university of hong kong" are two named entities, allowing us to fix the wrong name boundaries of the English NER.

2.2 Bad translations

There are cases where the MT system does not recognize there is a name and translates it as something else, and if we do not refer to the source language, we sometimes cannot understand the sentence, or annotate it.

Ex. 3. xinhua shanghai , january 1 (<ORG>**feng yizhen su lofty**</ORG>) snow , frozen , and the shanghai airport staff in snow and inalienable .

The translation system does not output the names correctly, and only when we look at the Chinese sentence can we know that there are two person names here, one is "feng yizhen", and the other is "su lofty", where the second one is translated incorrectly. English NER treats the whole as an ORGANIZATION as there is no punctuation to separate the two names.

2.3 Unknown foreign names

There are many Chinese GPE and PERSON names which are missed because they appear rarely in English text, especially city, county or even province names, and so are hard for English NER to detect or classify. However, on the Chinese side, they may be common names and so easily tagged.

[1] The MT system we used generates monocase translations, so we show all the translations in lower case.

[2] We use the entity types of ACE (the Automatic Content Extraction evaluation) for name types. Here ORG = "ORGANIZATION" is the tag for an organization; GPE = "Geo-Political Entity" is the tag for a location with a government; other locations (e.g., "Sahara Desert") are tagged as LOCATION.

Ex. 4. At present, **shishi city** in the province to achieve a village public transportation, village water ; village of cable television .

The city names in examples 4 are famous in Chinese but do not appear much in English text, and so are missed by the English NER; however, a Chinese NER would be able to tag them as named entities.

3 Entity Translation System

The MT pipeline we employ begins with an Entity Translation (ET) system which identifies and translates the names in the text (Heng Ji et al., 2007). This system runs a source-language NER (based on an HMM) and then uses a variety of strategies to translate the names it identifies. One strategy, for example, uses a corpus-trained name transliteration component coupled with a target language model to select the best transliteration. The source text, annotated with name translations, is then passed to a statistical, phrase-based MT system (Zens and Ney, 2004). Depending on its phrase table and language model, this name-aware MT system would decide whether to accept the translation provided by ET. Experiments show that the MT system with ET pre-processing can produce better translations than the MT system alone, with 17% relative error reduction on overall name translation.

The strategy combining multiple transliterations and selection based on a language model is particularly effective for foreign (non-Chinese) person names rendered in Chinese. If these names did not appear in the bilingual training material, they would be mistranslated by an MT system without ET. These names are often also difficult for the English tagger, so ET can benefit both translation and name recognition.

For each name tagged by ET, we see if the translation string proposed by ET appears in the translation produced by the MT system. If so, we use the ET output to assign an 'ET name type' to that string in the translation. This approach avoids the problems of using word alignments from the MT system; in particular, the alignment of function words in English with names in Chinese.

4 Integrating source and target information

We first try a very simple merge method to see how much gain can be gotten by simply combining the two sources. After that, we describe a corpus-trained model which addresses some of the tag conflict situations and gets additional gains.

4.1 Results from English NER and ET

First, we analyzed the English NER and ET output to see the named entity distribution of the two sources. We focus on the differences between them because when they agree, we can expect little improvement from using source language information. In the nist05 data, we find 1893 named entities in the English NER output (target language part) and 1968 named entities in the ET output (source language part); 1171 of them are the same. This means that 38.14% of the names tagged in the target language and 40.5% of those in the source language do not have a corresponding tag in the other language, which suggests that the source and target NER may have different strengths on name tagging.

We checked the names which are tagged differently, and there are 347 correct names from ET missed by English NER and 418 from English NER missed by ET.

4.2 Simple Merge

First, in order to see if the ET system can really help the English NER, we do a simple merge experiment, which just adds the named entities extracted from the ET system into the English NER results, so long as there is no conflict between them (i.e., so long as the ET-tagged name does not overlap an English NE-tagged name).

Our experiments show that this simple method can improve the English NER result substantially (Table 5-1), especially for recall, confirming our intuition.

We checked the errors produced by this simple merge method, and divided them into four types.

1. Missed by both sources.
2. Missed by one source and erroneously tagged by the other
3. Erroneously tagged by both sources
4. Conflict situations where the English NE-tagged name is wrong but the ET-tagged name is correct.

Although there is not much we can do for the first three error types, we can address the last error type by some intelligent learning method. In NIST05 data, there are 261 names which have conflicts, and we can get more gains here.

There are two kinds of conflicts: A type conflict which occurs when the ET and English NER tag the same named entity but give it different types; and a boundary conflict which occurs when there is a tag overlap between English NER and ET. We treat these two kinds of conflict differently by using different features to indicate them.

4.3 Maximum Entropy Markov Model

We use a MEMM (Maximum Entropy Markov Model) as our tagging model. An MEMM is a variation on traditional Hidden Markov Models (HMM). Like an HMM, it attempts to characterize a string of tokens as a most likely set of transitions through a Markov model. The MEMM allows observations to be represented as arbitrary overlapping features (such as word, capitalization, formatting, part-of-speech), and defines the conditional probability of state sequences given observation sequences. It does this by using the maximum entropy framework to fit a set of exponential models that represent the probability of a state given an observation and the previous state (McCallum et al. 2000).

In our experiment, we train the maximum entropy framework at the token level, and use the BIO types as the states to be predicted. There are four entity types: PERSON, ORGANIZATION, GPE and LOCATION, and so a total of 9 states.

4.4 Feature Sets for MEMM

In our experiment, we are interested not only in training a module, but also in measuring the different performance for different scales of training corpora. If a small annotated corpus can get reasonable gain, this method for combining taggers will be much more practical.

As a result, we first build a small feature set and enlarge it by adding more features, expecting that the small feature set may get better performance with a small training corpus.

Set 1: Features Focusing on Current Tag and Previous State Information

We first try to use few features to see how much gain we can get if we only consider the tag information from ET and English NER, and the previous state. These features are:

F1: current token's type in ET
F2: current token's type in English NER
F3: Feature1+Feature2
F4: if there is a type conflict + ET type + English NER type
F5: if there is a type conflict +ET type confidence + English NER confidence
F6: if there is a boundary conflict + ET type + English NER type
F7: if there is a boundary conflict + ET token confidence + English NER confidence
F8: state for the previous token

F4 and F5 are used to help resolve the type conflicts, and F6 and F7 to resolve boundary conflicts. When there is a conflict, we need the confidence information from both ET and English NER to indicate which side to choose.

The English NER reports a *margin*, which can be used to gauge tag confidence. The margin is the difference in log probability between the top tagging hypothesis and a hypothesis which assigns the name a different NE tag, or no NE tag. We use this as the confidence of English NER output.

For ET output, the situation is more complicated. We use different confidence methods for type and boundary conflicts. For type conflicts, we use the source of the ET translation as the "type confidence", for example, if the ET result comes from a person name list, the output is probably correct. For boundary conflicts, as the ET system uses some pruning strategy to fix the boundary errors in word alignment, and the translation procedure contains several disparate components which produce different kind of confidence measure, it is not reasonable to use Chinese NER confidence as the confidence estimate. As a result, we check if the token is capitalized in ET translation, and treat it as the "token confidence".

Set 2: Set 1 + Current Token Information

F9: current token + ET type+ English NER type

Token information can be used to predict the result when there is a conflict, as the conflict reason varies and in some cases without knowing the token itself, it is hard to know the right choice. As a result, we add the current token feature but this is the only place we use token information.

Set 3: Set2 + Sequence Information

Our experiments showed some performance gain with only the current token features and the previous state, but we still wanted to see if additional features – such as information on the previous and following tokens – would help. To this end, we added such features, while still retaining our focus on the ET and English NER information:

F10: English NER result of the current token + that of the previous token

F11: ET result of the current token + ET result of the previous token.

F12: English NER result of the current token + that of the next token.

F13: ET result of the current token + that of the next token.

5 Experiment

The experiment was carried out on the Chinese part of the NIST 05 machine translation evaluation (NIST05) and NIST 04 machine translation evaluation (NIST04) data, where NISTT05 contains 100 documents and NIST04 contains 200 documents. We annotated all the data in NIST05 and 120 documents for NIST04 for our experiment.

The ET system used a Chinese HMM-based NER trained on 1,460,648 words; the English name tagger was also HMM-based and trained on 450,000 words.

First, we want to see the result with very small training data, and so divided the NIST05 data into 5 subsets, each containing 20 documents. We ran a cross validation experiment on this small corpus, with 4 subsets as training data and 1 as testing data. We refer to this configuration as Corpus1[3].

Second, to see whether increasing the training data would appreciably influence the result, we added the annotated NIST04 data into the training corpus, and we call this configuration Corpus2.

[3] We conducted some experiments with a small corpus in which we relied on the alignment information from the MT system, but the results were much worse than using the ET output. Simple merge using alignment yielded a name tagger F score of 73.34% (1.42% worse than the baseline, 75.76%), while ET F score of 81.23%; MEMM with minimal features using alignment yielded an improvement of 1.7% (vs. 7.9% using ET).

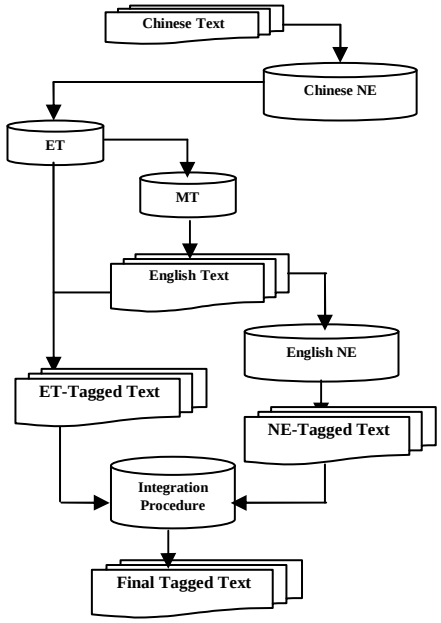


Figure 1. Flow chart of our system

5.1 Simple Merge Result

The simple merge method gets a significant F-measure gain of 7.15% from the English NER baseline, which confirms our intuition that some named entities are easy to tag in source language and others in target language. This represents primarily a significant recall improvement, 14.37%.

	NER baseline	Simple Merge
P	85.68	82.70
R	64.39	78.76
F	73.53	**80.68**

Table 1. Simple merge method on Corpus1 (100 documents)

5.2 Integrating Results on Corpus1

On this small training corpus, we test each subset with other subsets as training data, and calculate the total performance on the whole corpus. The best result comes from Set2 instead of Set3, presumably because the training data is too small to handle the richer model of Set3. Our experiment shows that we can get 1.9% gain over simple merge method with Set 2 using 80 documents as training data.

	Simple Merge	Set1	Set2	Set3
P	82.70	84.73	84.72	84.48
R	78.76	78.01	80.55	80.15
F	80.68	81.23	**82.58**	82.26

Table 2. Results on Corpus1, which contains 100 documents, with 80 documents used for training at each fold.

5.3 Integrating Results on Corpus2

On this corpus, every training data set contains 200 documents, and we can get a gain of 2.74% over the simple merge method. With the larger training set, the richer model (Set 3) now outperforms the others.

	Simple Merge	Set1	Set2	Set3
P	82.70	85.04	85.15	85.78
R	78.76	78.09	80.59	81.18
F	80.68	81.42	82.81	**83.42**

Table 3. Result on Corpus2 (220 documents), with 200 documents used for training at each fold of cross-validation.

On corpus2, Using a Wilcoxon Matched-Pairs test, with a 10-fold division, all the sets perform significantly better (in F-measure) than the simple merge at a 95% confidence level.

6 Prior Work

Huang and Vogel (2002) describe an approach to extract a named entity translation dictionary from a bilingual corpus while concurrently improving the named entity annotation quality. They use a statistical alignment model to align the entities and iteratively extract the name pairs with higher alignment probability and treat them as global information to improve the monolingual named entity annotation quality for both languages. Using this iterative method, they get a smaller but cleaner named entity translation dictionary and improve the annotation F-measure from 70.03 to 78.15 for Chinese and 73.38 to 81.46 in English. This work is similar in using information from the source language (in this case mediated by the word alignment) to improve the target language tagging. However, they used bi-texts (with hand-translated, relatively high-quality English) and so did not encounter the problems, mentioned above, which arise with MT output.

7 Conclusion

We present an integrated approach to extract the named entities from machine translated text, using name entity information from both source and target language. Our experiments show that with a combination of ET and English NER, we can get a considerably better NER result than would be possible with either alone, and in particular, a large improvement in name identification recall.

MT output poses a challenge for any type of language analysis, such as relation or event recognition or predicate-argument analysis. Even though MT is improving, this problem is likely to be with us for some time. The work reported here indicates how source language information can be brought to bear on such tasks.

The best F-measure in our experiments exceeds the score of the English NER on reference text, which reflects the intuition that even for well translated text, we can still benefit from source language information.

Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency under Contract No. HR0011-06-C-0023, and the National Science Foundation under Grant NO. IIS-0534700. Any opinions, findings and conclusions expressed in this material are those of the author and do not necessarily reflect the views of the U. S. Government.

References

Yonggang Deng, Byrne and William J. 2005. *HMM Word and Phrase Alignment for Statistical Machine Translation.* Proc. Human Language Technology Conference and Empirical Methods in Natural Language Processing**.**

Fei Huang and Vogel, S. 2002. *Improved named entity translation and bilingual named entity extraction.* Proc. Fourth IEEE Int'l. Conf. on Multimodal Interfaces.

A. McCallum, D. Freitag and F. Pereira. 2000. *Maximum entropy Markov models for information extraction and segmentation.* Proc. 17th International Conf. on Machine Learning.

Heng Ji, Matthias Blume, Dayne Freitag,Ralph Grishman, Shahram Khadivi and Richard Zens. 2007. *NYU-Fair Isaac-RWTH Chinese to English Entity Translation 07 System.* Proceedings of ACE ET 2007 PI/Evaluation Workshop. Washington.

Richard Zens and Hermann Ney. 2004. *Improvements in phrase-based statistical Machine Translation.* In Proc. HLT/NAACL, Boston

A Re-examination on Features in Regression Based Approach to Automatic MT Evaluation

Shuqi Sun, Yin Chen and Jufeng Li
School of Computer Science and Technology
Harbin Institute of Technology, Harbin, China
{sqsun, chenyin, jfli}@mtlab.hit.edu.cn

Abstract

Machine learning methods have been extensively employed in developing MT evaluation metrics and several studies show that it can help to achieve a better correlation with human assessments. Adopting the regression SVM framework, this paper discusses the linguistic motivated feature formulation strategy. We argue that "blind" combination of available features does not yield a general metrics with high correlation rate with human assessments. Instead, certain simple intuitive features serve better in establishing the regression SVM evaluation model. With six features selected, we show evidences to support our view through a few experiments in this paper.

1 Introduction

The automatic evaluation of machine translation (MT) system has become a hot research issue in MT circle. Compared with the huge amount of manpower cost and time cost of human evaluation, the automatic evaluations have lower cost and reusability. Although the automatic evaluation metrics have succeeded in the system level, there are still on-going investigations to get reference translation better (Russo-Lassner et al., 2005) or to deal with sub-document level evaluation (Kulesza et al., 2004; Leusch et al, 2006).

N-grams' co-occurrence based metrics such as BLEU and NIST can reach a fairly good correlation with human judgments, but due to their consideration for the capability of generalization across multiple languages, they discard the inherent linguistic knowledge of the sentence evaluated. Actually, for a certain target language, one could exploit this knowledge to help us developing a more "human-like" metric. Giménez and Márquez (2007) showed that compared with metrics limited in lexical dimension, metrics integrating deep linguistic information will be more reliable.

The introduction of machine learning methods aimed at the improvement of MT evaluation metrics' precision is a recent trend. Corston-Oliver et al. (2001) treated the evaluation of MT outputs as classification problem between human translation and machine translation. Kulesza et al. (2004) proposed a SVM classifier based on *confidence score*, which takes the distance between feature vector and the decision surface as the measure of the MT system's output. Joshua S. Albrecht et al. (2007) adopted regression SVM to improve the evaluation metric.

In the rest of this paper, we will first discuss some pitfalls of the n-gram based metrics such as BLEU and NIST, together with the intuition that factors from the linguist knowledge can be used to evaluate MT system's outputs. Then, we will propose a MT evaluation metric based on SVM regression using information from various linguistic levels (lexical level, phrase level, syntax level and sentence-level) as features. Finally, from empirical studies, we will show that this metric, with less simple linguistic motivated features, will result in a better correlation with human judgments than previous regression-based methods.

2 N-gram Based vs Linguistic Motivated Metrics

N-gram co-occurrence based metrics is the main trend of MT evaluation. The basic idea is to compute the similarity between MT system output and

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 25–30,
Columbus, June 2008.

several human reference translations through the co-occurrence of n-grams. BLEU (Papineni et al., 2002) is one of the most popular automatic evaluation metrics currently used. Although with a good correlation with human judgment, it still has some defects:

● BLEU considers precision regardless of recall. To avoid a low recall, BLEU introduces a *brevity penalty* factor, but this is only an approximation.

● Though BLEU makes use of high order n-grams to assess the fluency of a sentence, it does not exploit information from inherent structures of a sentence.

● BLEU is a "perfect matching only" metric. This is a serious problem. Although it can be alleviated by adding more human reference translations, there may be still a number of informative words that will be labeled as "unmatched".

● BLEU lacks models determining each n-gram's own contribution to the meaning of the sentence. Correct translations of the headwords which express should be attached more importance to than that of accessory words e.g.

● While computing geometric average of precisions from unigram to n-gram, if a certain precision is zero, the whole score will be zero.

In the evaluation task of a MT system with certain target language, the intuition is that we can fully exploit linguistic information, making the evaluation progress more "human-like" while leaving the capability of generalization across multiple languages (just the case that BLEU considers) out of account.

Following this intuition, from the plentiful linguist information, we take the following factors in to consideration:

● Content words are important to the semantic meaning of a sentence. A better translation will include more substantives translated from the source sentence than worse ones. In a similar way, a machine translation should be considered a better one, if more content words in human reference translations are included in it.

● At the phrase level, the situation above remains the same, and what is more, real phrases are used to measure the quality of the machine translations instead of merely using n-grams which are of little semantic information.

● In addition, the length of translation is usually in good proportion to the source language. We believe that a human reference translation sentence has a moderate byte-length ratio to the source sentence. So a machine translation will be depreciated if it has a ratio considerably different from the ratio calculated from reference sentences.

● Finally, a good translation must be a "well-formed" sentence, which usually brings a high probability score in language models, e.g. n-gram model.

In the next section, using regression SVM, we will build a MT evaluation metric for Chinese-English translation with features selected from above aspects.

3 A Regression SVM Approach Based on Linguistic Motivated Features

Introducing machine learning methods to establish MT evaluation metric is a recent trend. Provided that we could get many factors of human judgments, machine learning will be a good method to combine these factors together. As proved in the recent literature, learning from regression is of a better quality than from classifier (Albrecht and Hwa, 2007; Russo-Lassner et al., 2005; Quirk, 2004). In this paper, we choose regression support vector machine (SVM) as the learning model.

3.1 Learning from human assessment data

The machine translated sentences for model training are provided with human assessment data score together with several human references. Each sentence is treated as a training example. We extract feature vectors from training examples, and human assessment score will act as the output of the target function. The regression SVM will generate an approximated function which maps multi-dimensional feature vectors to a continuous real value with a minimal error rate according to a loss function. This value is the result of the evaluation process.

Figure 1 shows our general framework for regression based learning, in which we train the SVM with a number of sentences $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, … with human assessment scores y_1, y_2, … and use the trained model to evaluate an test sentence $\boldsymbol{x}$ with feature vector $(f_1, f_2, ..., f_n)$. To determine which indicators of a sentence are chosen as features is research in progress, but we contend that "the more features, the better quality" is not always true. Large feature sets require more computation cost, though maybe result in a metric with a better corre-

lation with human judgments, it can also be achieved by introducing a much smaller feature set. Moreover, features may conflict with each others, and bring down the performance of the metric. We will show this in the next section, using less than 10 features stated in section 3.2. Some details of the implementation will also be described.

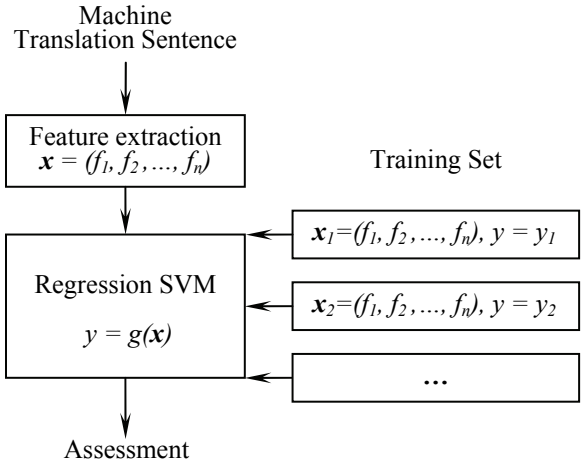


Figure 1: SVM based model of automatic MT evaluation metric

3.2 Feature selection

A great deal of information can be extracted from the MT systems' output using linguistic knowledge. Some of them can be very informative while easy to obtain.

As considered in section 2, we choose factors from lexical level, phrase level, syntax level and sentence-level as features to train the SVM.

● Features based on translation quality of content words

The motivation is that content words are carrying more important information of a sentence compared with function words. In this paper, content words include nouns, verbs, adjectives, adverbials, pronouns and cardinal numerals. The corresponding features are the precision of content words defined in Eq. 1 and the recall defined in Eq. 2 where *ref* means reference translation.

$$precision_{con}(t) = \frac{\#correctly\_translated\_cons\_in\_t}{\#cons\_in\_t} \quad (1)$$

$$recall_{con}(t) = \frac{\#cons\_in\_ref\_correctly\_translated\_in\_t}{\#cons\_in\_the\_ref} \quad (2)$$

● Features based on cognate words matching

English words have plenty of morphological changes. So if a machine translation sentence shares with a human reference sentence some cognates, it contains at least some basic information correct. And if we look at it in another way, words that do not match in the original text maybe match after morphological reduction. Thus, differences between poor translations will be revealed. Similarly, we here define the content word precision and recall after morphological reduction in Eq. 3 and Eq. 4 where *mr_cons* means content words after morphological reduction:

$$precision_{mr\_con}(t) = \frac{\#correctly\_translated\_mr\_cons\_in\_t}{\#mr\_cons\_in\_t} \quad (3)$$

$$recall_{mr\_con}(t) = \frac{\#mr\_cons\_in\_ref\_correctly\_translated\_in\_t}{\#mr\_cons\_in\_the\_ref} \quad (4)$$

● Features based on translation quality of phrases

Phrases are baring the weight of semantic information more than words. In manual evaluation, or rather, in a human's mind, phrases are paid special attention to. Here we parse every sentence[1] and extract several types of phrases, then, compute the precision and recall of each type of phrase according to Eq. 5 and Eq. 6[2]:

$$precision_{phr}(t) = \frac{\#correctly\_translated\_phrs\_in\_t}{\#phrs\_in\_t} \quad (5)$$

$$recall_{phr}(t) = \frac{\#phr\_in\_ref\_correctly\_translated\_in\_t}{\#phr\_in\_the\_ref} \quad (6)$$

In practice, we found that if we compute these two indicators by matching phrases case-insensitive, we will receive a metric with higher performance. We speculate that by doing this the difference between poor translations is revealed just like morphological reduction.

● Features based on byte-length ratio

Gale and Church (1991) noted that he byte-length ratio of target sentence to source sentence is normally distributed. We employ this observation by computing the ratio of reference sentences to

[1] The parser we used is proposed by Michael Collins in Collins (1999).

[2] Only precision and recall of NP are used so far. Other types of phrase will be added in future study.

source sentences, and then calculating the mean *c* and variance *s* of this ratio. So if we take the ratio *r* as a random variable, *(r-c)/s* has a normal distribution with mean 0 and variance 1. Then we compute the same ratio of machine translation sentence to source sentence, and take the output of p-norm function as a feature:

$$f(t) = P_{norm}(\frac{lenght\_of\_t / length\_of\_src - c}{s}) \quad (7)$$

● Features based on parse score

The usual practice to model the "well-formedness" of a sentence is to employ the n-gram language model or compute the syntactic structure similarity (Liu and Gildea 2005). However, the language model is widely adopted in MT, resulting less discrimination power. And the present parser is still not satisfactory, leading much noise in parse structure matching.

To avoid these pitfalls in using LM and parser, here we notice that the score of a parse by the parser also reflects the quality of a sentence. It may be regarded as a syntactic based language model score as well as an approximate representation of parse structure. Here we introduce the feature based on parser's score as:

$$paser\_score(t) = -\frac{100}{mark\_of\_t\_given\_by\_parser} \quad (8)$$

4 Experiments

We use SVM-Light (Joachims 1999) to train our learning models. Our main dataset is NIST's 2003 Chinese MT evaluations. There are 6×919=5514 sentences generated by six systems together with human assessment data which contains a fluency score and adequacy score marked by two human judges. Because there is bias in the distributions of the two judges' assessment, we normalize the scores following Blatz et al. (2003). The normalized score is the average of the sum of the normalized fluency score and the normalized adequacy score.

To determine the quality of a metric, we use Spearman rank correlation coefficient which is distribution-independent between the score given to the evaluative data and human assessment data. The Spearman coefficient is a real number ranging from -1 to +1, indicating perfect negative correlations or perfect positive correlations. We take the correlation rates of the metrics reported in Albrecht and Hwa (2007) and a standard automatic metric BLEU as a baseline comparison.

Among the features described in section 3.2, we finally adopted 6 features:

● Content words precision and recall after morphological reduction defined in Eq. 3 and Eq. 4.

● Noun-phrases' case insensitive precision and recall.

● P-norm (Eq. 7) function's output.

● Rescaled parser score defined in Eq. 8. Our first experiment will compare the correlation rate between metric using rescaled parser score and that using parser score directly.

4.1 Different kernels

Intuitively, features and the resulting assessment are not in a linear correlation. We trained two SVM, one with linear kernel and the other with Gaussian kernel, using NIST 2003 Chinese dataset. Then we apply the two metrics on NIST 2002 Chinese Evaluation dataset which has 3×878=2634 sentences (3 systems total). The results are summarized in Table 1. For comparison, the result from BLEU is also included.

Feature	Linear	Gaussian	BLEU
Rescale	0.320	**0.329**	0.244
Direct	0.317	0.224	

Table 1: Spearman rank-correlation coefficients for regression based metrics using linear and Gaussian kernel, and using rescaled parser score or directly the parser score. Coefficient for BLEU is also involved.

Table 1 shows that the metric with Gaussian kernel using rescaled parser score gains the highest correlation rate. That is to say, Gaussian kernel function can capture characteristics of the relation better, and rescaling the parser score can help to increase the correlation with human judgments. Moreover, as other features range from 0 to 1, we can discover in the second row of Table 1 that Gaussian kernel is suffering more seriously from the parser score which is ranging distinctly. In following experiments, we will adopt Gaussian kernel to train the SVM and rescaled parser score as a feature.

4.2 Comparisons within the year 2003

We held out 1/6 of the assessment dataset for parameter turning, and on the other 5/6 of dataset, we perform a five-fold cross validation to verify the metric's performance. In comparison we introduce

several metrics' coefficients reported in Albrecht and Hwa (2007) including smoothed BLEU (Lin and Och, 2004), METEOR (Banerjee and Lavie, 2005), HWCM (Liu and Gildea 2005), and the metric proposed in Albrecht and Hwa (2007) using the full feature set. The results are summarized in Table 2:

Metric	Coefficient
Our Metric	0.515
Albrecht, 2007	**0.520**
Smoothed BLEU	0.272
METEOR	0.318
HWCM	0.288

Table 2: Comparison among various metrics. Learning-based metrics are developed from NIST 2003 Chinese Evaluation dataset and tested under five-fold cross validation.

Compared with reference based metrics such as BLEU, the regression based metrics yield a higher correlation rate. Generally speaking, for a given source sentence, there is usually a lot of feasible translations, but reference translations are always limited though this can be eased by adding references. On the other hand, regression based metrics is independent of references and make the assessment by mapping features to the score, so it can make a better judgment even dealing with a translation that doesn't match the reference well.

We can also see that our metric which uses only 6 features can reach a pretty high correlation rate which is close to the metric proposed in Albrecht and Hwa (2007) using 53 features. That confirms our speculation that a small feature set can also result in a metric having a good correlation with human judgments.

4.3 Crossing years

Though the training set and test set in the experiment described above are not overlapping, in the last, they come from the same dataset (NIST 2003). The content of this dataset are Xinhua news and AFC news from Jan. 2003 to Feb. 2003 which has an inherent correlation. To test the capability of generalization of our metric, we trained a metric on the whole NIST 2003 Chinese dataset (20% data are held out for parameter tuning) and applied it onto NIST 2002 Chinese Evaluation dataset. We use the same metrics introduced in section 4.2 for comparison. The results are summarized in Table 3:

Metric	Coefficient
Our Metric	**0.329**
Albrecht, 2007	0.309
Smoothed BLEU	0.269
METEOR	0.290
HWCM	0.260

Table 3: Cross year experiment result. All the learning based metrics are developed from NIST 2003.

The content of NIST 2002 Chinese dataset is Xinhua news and Zaobao's online news from Mar. 2002 to Apr. 2002. The most remarkable characteristic of news is its timeliness. News come from the year 2002 are nearly totally unrelated to that from the year 2003. It can be seen from Table 3 that we have got the expected results. Our metric can generalize well across years and yields a better correlation with human judgments.

4.4 Discussions

Albrecht and Hwa (2007) and this paper both adopted a regression-based learning method. In fact, the preliminary experiment is strictly set according to their paper. The most distinguishing difference is that the features in Albrecht and Hwa (2007) are collections of existing automatic evaluation metrics. The total 53 features are computationally heavy (for the features from METEOR, ROUGE, HWCM and STM). In comparison, our metric made use of six features coming from linguistic knowledge which can be easily obtained. Moreover, the experiments show that our metric can reach a correlation with human judgments nearly as good as the metric described in Albrecht and Hwa (2007), with a much lower computation cost. And when we applied it to a different year's dataset, its correlation rate is much better than that of the metric from Albrecht and Hwa (2007), showing us a good capability of generalization.

To account for this, we deem that the regression model is not resistant to data overfiting. If provided too much cross-dependent features for a limited training data, the model is prone to a less generalized result. But, it is difficult in practice to locate those key features in human perception of translation quality because we are lack of explicit evidences on what human actually use in translation evaluation. In such cases, this paper uses only "simple feature in key linguistic aspects", which reduces the risk of overfitting and bring a more generalized regression results.

Compared with the literature, the "byte-length ratio between source and translation" and the "parse score" are original in automatic MT evaluation modeling. The parse score is proved to be a good alternative to LM. And it helps to avoid the errors of parser in parse structure (the experiment to verify this claim is still on-going).

It should be noted that feature selection is accomplished by empirically exhaustive test on the combination of the candidate features. In future work, we will test if this strategy will help to get better results for MT evaluation, e.g. try-on the selection between the 53 features in Albrecht and Hwa (2007). And, we will also test to see if linguistic motivated feature augmentation would bring further benefit.

5 Conclusion

For the metrics based on regressing, it is not always true that more features and complex features will help in performance. If we choose features elaborately, simple features are also effective. In this paper we proposed a regression based metric with a considerably small feature set that yield performance of the same level to the metrics with a large set of 53 features. And the experiment of the cross-year validation proves that our metric bring a more generalized evaluation results by correlating with human judgments better.

Acknowledgements

This research is support by Natural Science Foundation of China (Grant No. 60773066) and National 863 Project (Grant No. 2006AA01Z150)

References

Joshua S. Albrecht and Rebecca Hwa. 2007. A Re-examination of Machine Learning Approaches for Sentence-Level MT Evaluation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics* , pages 880-887, Prague, Czech Republic, June.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the Association for Computational Linguistics Conference* 2005: 65-73. Ann Arbor, Michigan.

John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2003. Confidence estimation for machine translation. In *Technical Report Natural Language Engineering Workshop Final Report*, pages 97-100, Johns Hopkins University.

Simon Corston-Oliver, Michael Gamon, and Chris Brockett. 2001. A machine learning approach to the automatic evaluation of machine translation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 140-147, Toulouse, France, July.

W. Gale and K. W. Church. 1991. A Program for Aligning Sentences in Bilingual Corpora. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 177-184, Berkeley.

Jesús Giménez and Lluís Màrquez. 2007. Linguistic Features for Automatic Evaluation of Heterogenous MT Systems. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 256-264, Prague, Czech Republic, June.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In Bernhard Schöelkopf, Christopher Burges, and Alexander Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press.

Alex Kulesza and Stuart M. Shieber. 2004. A learning approach to improving sentence-level MT evaluation. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*, pages 75-84, Baltimore, MD, October.

Gregor Leusch, Nicola Ueffing, and Hermann Ney. 2006. CDER: Efficient MT evaluation using block movements. In *The Proceedings of the Thirteenth Conference of the European Chapter of the Association for Computational Linguistics*, pages 241-248.

Chin-Yew Lin & Franz Josef Och. 2004. Automatic Evaluation of Machine Translation Quality Using Longest Common Subsequence and Skip-Bigram Statistics. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, pages 606-613, Barcelona, Spain, July.

Ding Liu and Daniel Gildea. 2005. Syntactic features for evaluation of machine translation. In *ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 25-32, June.

Christopher B. Quirk. 2004. Training a Sentence-Level Machine Translation Confidence Measure, In *Proceedings of LREC 2004*, pages 825-828.

Grazia Russo-Lassner, Jimmy Lin, and Philip Resnik. 2005. A Paraphrase-Based Approach to Machine Translation Evaluation. In *Technical Report LAMP-TR-125/CS-TR-4754/UMIACS-TR-2005-57*, University of Maryland, College Park, August.

The role of positive feedback in Intelligent Tutoring Systems

Davide Fossati
Department of Computer Science
University of Illinois at Chicago
Chicago, IL, USA
dfossa1@uic.edu

Abstract

The focus of this study is positive feedback in one-on-one tutoring, its computational modeling, and its application to the design of more effective Intelligent Tutoring Systems. A data collection of tutoring sessions in the domain of basic Computer Science data structures has been carried out. A methodology based on multiple regression is proposed, and some preliminary results are presented. A prototype Intelligent Tutoring System on linked lists has been developed and deployed in a college-level Computer Science class.

1 Introduction

One-on-one tutoring has been shown to be a very effective form of instruction (Bloom, 1984). The research community is working on discovering the characteristics of tutoring. One of the goals is to understand the strategies tutors use, in order to design effective learning environments and tools to support learning. Among the tools, particular attention is given to Intelligent Tutoring Systems (ITSs), which are sophisticated software systems that can provide personalized instruction to students, in some respect similar to one-on-one tutoring (Beck et al., 1996). Many of these systems have been shown to be very effective (Evens and Michael, 2006; Van Lehn et al., 2005; Di Eugenio et al., 2005; Mitrović et al., 2004; Person et al., 2001). In many experiments, ITSs induced learning gains higher than those measured in a classroom environment, but lower than those obtained with one-on-one interactions with human tutors. The belief of the research community is that knowing more about human tutoring would help improve the design of ITSs. In particular, the effective use of natural language might be a key element. In most of the studies mentioned above, systems with more sophisticated language interfaces performed better than other experimental conditions.

An important form of student-tutor interaction is *feedback*. *Negative feedback* can be provided by the tutor in response to students' mistakes. An effective use of negative feedback can help the student correct a mistake and prevent him/her from repeating the same or a similar mistake again, effectively providing a learning opportunity to the student. *Positive feedback* is usually provided in response to some correct input from the student. Positive feedback can help students reinforce the correct knowledge they already have, or successfully integrate new knowledge, if the correct input provided by the student was originated by a random or tentative step.

The goal of this study is to assess the relevance of positive feedback in tutoring, and build a computational model of positive feedback that can be implemented in ITSs. Even though some form of positive feedback is present in many successful ITSs, the predominant type of feedback generated by those systems is negative feedback, as those systems are designed to react to students mistakes. To date, there is no systematic study of the role of positive feedback in ITSs in the literature. However, there is an increasing amount of evidence that suggests that positive feedback may be very important in enhancing students' learning. In a detailed study in a controlled environment and domain, the letter pattern extrapolation task, Corrigan-Halpern (2006) found

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 31–36,
Columbus, June 2008.

that subjects given positive feedback performed better in an assessment task than subjects receiving negative feedback. In another study on the same domain, Lu (2007) found that the ratio of the positive over negative messages in her corpus of expert tutoring dialogues is about 4 to 1, and the ratio is even higher in the messages presented by her successful ITS modeled after an expert tutor, being about 10 to 1. In the dataset subject of this study, which is on a completely different domain —Computer Science data structures— such a high ratio of positive over negative feedback messages still holds, in the order of about 8 to 1. In a recent study, Barrow et al. (2008) showed that a version of their SQL-Tutor enriched with positive feedback generation helped students learn faster than another version of the same system delivering negative feedback only.

What might be the educational value of positive feedback in ITSs? First of all, positive feedback may be an effective motivational technique (Lepper et al., 1997). Positive feedback can also have cognitive value. In a problem solving setting, the student can make a tentative (maybe random) step towards the correct solution. At this point, positive feedback from the tutor may be important in helping the student consolidate this step and learn from it. Some researchers outlined the importance of self-explanation in learning (Chi, 1996; Renkl, 2002). Positive feedback has the potential to improve self-explanation, in terms of quantity and effectiveness. Another issue is how students perceive and accept feedback (Weaver, 2006), and, in the case of automated tutoring systems, whether students read feedback messages at all (Heift, 2001). Positive feedback might also make students more willing to accept help and advice from the tutor.

2 A study of human tutoring

The domain of this study is Computer Science data structures, specifically *linked lists*, *stacks*, and *binary search trees*. A corpus of 54 one-on-one tutoring sessions has been collected. Each individual student participated in only one tutoring session, with a tutor randomly assigned from a pool of two tutors. One of the tutors is an experienced Computer Science professor, with more than 30 years of teaching experience. The other tutor is a senior undergraduate student in Computer Science, with only one semester of previous tutoring experience. The tutoring sessions have been videotaped and transcribed. Student took a pre-test right before the tutoring session, and a post-test immediately after. An additional group of 53 students (control group) took the pre and post tests, but they did not participate in a tutoring session, and attended a lecture about a totally unrelated topic instead.

Topic	Tutor	Avg	Stdev	t	df	P
List	Novice	.09	.22	-2.00	23	.057
	Expert	.18	.26	-3.85	29	$< .01$
	Both	.14	.25	-4.24	53	$< .01$
	None	.01	.15	-0.56	52	ns
	iList	.09	.17	-3.04	32	$< .01$
Stack	Novice	.35	.25	-6.90	23	$< .01$
	Expert	.27	.22	-6.15	23	$< .01$
	Both	.31	.24	-9.20	47	$< .01$
	No	.05	.17	-2.15	52	$< .05$
Tree	Novice	.33	.26	-6.13	23	$< .01$
	Expert	.29	.23	-6.84	29	$< .01$
	Both	.30	.24	-9.23	53	$< .01$
	No	.04	.16	-1.78	52	ns

Table 1: Learning gains and t-test statistics

Paired samples t-tests revealed that post-test scores are *significantly higher* than pre-test scores in the two tutored conditions for all the topics, except for linked lists with the less experienced tutor, where the difference is only marginally significant. If the two tutored groups are aggregated, there is significant difference for all the topics. Students in the control group did *not* show significant learning for linked lists and binary search trees, and only marginally significant learning for stacks. Means, standard deviations, and t-test statistic values are reported in Table 1.

There is *no significant difference* between the two tutored conditions in terms of learning gain, expressed as the difference between post-score and pre-score. This is revealed by ANOVA between the two groups of students in the tutored condition. For lists, $F(1, 53) = 1.82$, $P = ns$. For stacks, $F(1, 47) = 1.35$, $P = ns$. For trees, $F(1, 53) = 0.32$, $P = ns$.

The learning gain of students that received tutoring is *significantly higher* than the learning gain of the students in the control group, for all the topics.

This is showed by ANOVA between the group of tutored students (with both tutors) and the control group. For lists, $F(1, 106) = 11.0$, $P < 0.01$. For stacks, $F(1, 100) = 41.4$, $P < 0.01$. For trees, $F(1, 106) = 43.9$, $P < 0.01$. Means and standard deviations are reported in Table 1.

3 Regression-based analysis

The distribution of scores across sessions shows a lot of variability (Table 1). In all the conditions, there are sessions with very high learning gains, and sessions with very low ones. This observation and the previous results suggest a new direction for subsequent analysis: instead of looking at the characteristics of a particular *tutor*, it is better to look at the features that discriminate the most successful *sessions* from the least successful ones. As advocated in (Ohlsson et al., 2007), a sensible way to do that is to adopt an approach based on multiple regression of learning outcomes per tutoring session onto the frequencies of the different features. The following analysis has been done adopting a hierarchical, linear regression model.

Prior knowledge First of all, we want to factor out the effect of *prior knowledge*, measured by the pre-test score. A linear regression model reveals strong effect of pre-test scores on learning gain (Table 2). However, the R^2 values show that there is a lot of variance left to be explained, especially for lists and stacks, although not so much for trees. Notice that the β weights are negative. That means students with higher pre-test scores learn *less* then students with lower pre-test scores. A possible explanation is that students with more previous knowledge have less *learning opportunity* than students with less previous knowledge.

Time on task Another variable that is recognized as important by the educational research community is *time on task*, and we can approximate it with the length of the tutoring session. In the hierarchical regression model, session length follows pre-test score. Surprisingly, session length has a significant effect only on linked lists (Table 2).

Student activity Another hypothesis is that the degree of *student activity*, in the sense of the amount of student's participation in the discussion, might relate to learning (Lepper et al., 1997; Chi et al., 2001). To test this hypothesis, the following definition of student activity has been adopted:

$$\text{student activity} = \frac{\text{\# of turns} - \text{\# of short turns}}{\text{session length}}$$

Turns are the sequences of uninterrupted speech of the student. *Short turns* are the student turns shorter than three words. The regression analysis revealed *no significant effect* of this measure of students' activity on learning gain.

Feedback The dataset has been manually annotated for *episodes* where positive or negative feedback is delivered. All the protocols have been annotated by one coder, and some of them have been double-coded by a second one (intercoder agreement: kappa = 0.67). Examples of feedback episodes are reported in Figure 1.

The number of positive feedback episodes and the number of negative feedback episodes have been introduced in the regression model (Table 2). The model showed a significant effect of feedback for linked lists and stacks, but no significant effect on trees. Interestingly, the effect of positive feedback is *positive*, but the effect of negative feedback is *negative*, as can be seen by the sign of the β value.

4 A tutoring system for linked lists

A new ITS in the domain of linked lists, *iList*, is being developed (Figure 2).

The iList system is based on the *constraint-based* design paradigm. Originally developed from a cognitive theory of how people might learn from performance errors (Ohlsson, 1996), constraint-based modeling has grown into a methodology used to build full-fledged ITSs, and an alternative to the model tracing approach adopted by many ITSs. In a constraint-based system, domain knowledge is modeled with a set of *constraints*, logic units composed of a *relevance condition* and a *satisfaction condition*. A constraint is irrelevant when the relevance condition is not satisfied; it is satisfied when both relevance and satisfaction conditions are satisfied; it is violated when the relevance condition is satisfied but the satisfaction condition is not. In the context of tutoring, constraints are matched against student

	T:	do you see a problem?
	T:	I have found the node a@l, see here I found the node b@l, and then I put g@l in after it.
Begin +	T:	here I have found the node a@l and now the link I have to change is +...
	S:	++ you have to link e@l <over xxx.> [>]
End +	T:	[<] <yeah> I have to go back to this one.
	S:	*mmhm
	T:	so I *uh once I'm here, this key is here, I can't go backwards.
Begin -	S:	<so you> [>] <you won't get the same> [//] would you get the same point out of writing t@l close to c@l at the top?
	T:	oh, t@l equals c@l.
	T:	no because you would have a type mismatch.
End -	T:	t@l <is a pointer> [//] is an address, and this is contents.

Figure 1: Positive and negative feedback (T = tutor, S = student)

Topic	Model	Predictor	β	R^2	P
List	1	Pre-test	-.45	.18	$< .05$
	2	Pre-test	-.40	.28	$< .05$
		Session length	.35		$< .05$
	3	Pre-test	-.35	.36	$< .05$
		Session length	.33		.05
		+ feedback	.46		.05
		- feedback	-.53		$< .05$
Stack	1	Pre-test	-.53	.26	$< .01$
	2	Pre-test	-.52	.24	$< .01$
		Session length	.05		ns
	3	Pre-test	-.58	.33	$< .01$
		Session length	.01		ns
		+ feedback	.61		$< .05$
		- feedback	-.55		$< .05$
Tree	1	Pre-test	-.79	.61	$< .01$
	2	Pre-test	-.78	.60	$< .01$
		Session length	.03		ns
	3	Pre-test	-.77	.59	$< .01$
		Session length	.04		ns
		+ feedback	.06		ns
		- feedback	-.12		ns
All	1	Pre-test	-.52	.26	$< .01$
	2	Pre-test	-.54	.29	$< .01$
		Session length	.20		$< .05$
	3	Pre-test	-.57	.32	$< .01$
		Session length	.16		.06
		+ feedback	.30		$< .05$
		- feedback	-.23		.05

Table 2: Linear regression

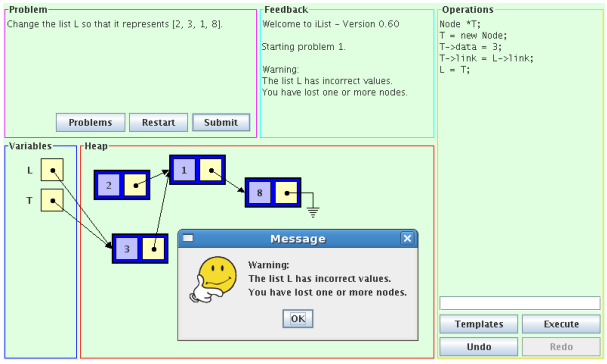


Figure 2: The iList system

solutions. Satisfied constraints correspond to knowledge that students have acquired, whereas violated constraints correspond to gaps or incorrect knowledge. An important feature is that there is no need for an explicit model of students' mistakes, as opposed to buggy rules in model tracing. The possible errors are implicitly specified as the possible ways in which constraints can be violated.

The architecture of iList includes a problem model, a constraint evaluator, a feedback manager, and a graphical user interface. Student model and pedagogical module, important components of a complete ITS (Beck et al., 1996), have not been implemented yet, and will be included in a future version. Currently, the system provides only simple negative feedback in response to students' mistakes, as customary in constraint-based ITSs.

A first version of the system has been deployed

into a Computer Science class of a partner institution. 33 students took a pre-test before using the system, and a post-test immediately afterwards. The students also filled in a questionnaire about their subjective impressions on the system. The interaction of the students with the system was logged.

T-test on test scores revealed that *students did learn* during the interaction with iList (Table 1). The learning gain is somewhere in between the one observed in the control condition and the one of the tutored condition. ANOVA revealed no significant difference between the control group and the iList group, nor between the iList group and the tutored group, whereas the difference between control and tutored groups is significant.

A preliminary analysis of the questionnaires revealed that students felt that iList helped them learn linked lists to a moderate degree (on a 1 to 5 scale: avg = 2.88, stdev = 1.18), but working with iList was interesting to them (avg = 4.0, stdev = 1.27). Students found the feedback provided by the system somewhat repetitive (avg = 3.88, stdev = 1.18), which is not surprising given the simple template-based generation mechanism. Also, the feedback was considered not very useful (avg = 2.31, 1.23), but at least not too misleading (avg = 2.22, stdev = 1.21). Interestingly, students declared that they read the feedback provided by the system (avg = 4.25, stdev = 1.05), but the logs of the system reveal just the opposite. In fact, on average, students read feedback messages for 3.56 seconds (stdev = 2.66 seconds), resulting in a reading speed of 532 words/minute (stdev = 224 words/minute). According to Carver's taxonomy (Carver, 1990), such speed indicates a quick skimming of the text, whereas reading for learning typically has a lower speed, in the order of 200 words/minute.

5 Future work

The main goal of this research is to build a computational model of positive feedback that can be used in ITSs. The study of empirical data and the system design and development will proceed in parallel, helping and informing each other as new results are obtained.

The conditions and the modalities of positive feedback delivery by tutors will be investigated from the human tutoring dataset. To do so, more coding categories will be defined, and the data will be annotated with these categories. The results of the statistical analysis over the first few coding categories will be used to guide the definition of more categories, that will be in turn used to annotate the data, and so on. An example of potential coding category is whether the student's action that triggered the feedback was prompted by the tutor or volunteered by the student. Another example is whether the feedback's content was a repetition of what the student just said or included additional explanation.

The first experiment with iList provided a comprehensive log of the students' interaction with the system. Additional analysis of this data will be important, especially because the nature of the interaction of a student with a computer system differs from the interaction with a human tutor. When working with a computer system, most of the interaction happens through a graphical interface, instead of natural language dialogue. Also, the interaction with a computer system is mostly student-driven, whereas our human protocols show a clear predominance of the tutor in the conversation. In the CS protocols, on average, 94% of the words belong to the tutor, and most of the tutors' discourse is some form of direct instruction. On the other hand, the interaction with the system will mostly consist of actions that students make to solve the problems that they will be asked to solve, with few interventions from the system. An interesting analysis that could be done on the logs is the discovery of sequential patterns using data mining algorithms, such as MS-GSP (Liu, 2006). Such patterns could then be regressed against learning outcomes, in order to assess their correlation with learning.

After the relevant features are discovered, a computational model of positive feedback will be built and integrated into iList. The model will encode knowledge extracted with machine learning approaches, and such knowledge will inform a discourse planner, responsible of organizing and generating appropriate positive feedback. The choiche of the specific machine learning and discourse planning methods will require extensive empirical investigation. Specifically, among the different machine learning methods, some are able to provide some sort of human-readable symbolic model, which can

be inspected to gain some insights on how the model works. Decision trees and association rules belong to this category. Other methods provide a less readable, black-box type of models, but they may be very useful and effective as well. Examples of such methods include Neural Networks and Markov Models. The ultimate goal of this research is to get both an effective model and to gain insights on tutoring. Thus, both classes of machine learning methods will be tried, with the goal of finding a balance between model effectiveness and model readability.

Finally, the system with enhanced feedback capabilities will be deployed and evaluated.

Acknowledgments

This work is supported by award N00014-07-1-0040 from the Office of Naval Research, and additionally by awards ALT-0536968 and IIS-0133123 from the National Science Foundation.

References

Devon Barrow, Antonija Mitrović, Stellan Ohlsson, and Michael Grimley. 2008. Assessing the impact of positive feedback in constraint-based tutors. In *ITS 2008, The 9th International Conference on Intelligent Tutoring Systems*, Montreal, Canada.

Joseph Beck, Mia Stern, and Erik Haugsjaa. 1996. Applications of AI in education. *ACM crossroads*. http://www.acm.org/crossroads/xrds3-1/aied.html.

B. S. Bloom. 1984. The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher*, 13:4–16.

Ronald P. Carver. 1990. *Reading Rate: A Review of Research and Theory*. Academic Press, San Diego, CA.

Michelene T.H. Chi, Stephanie A. Siler, Heisawn Jeong, Takashi Yamauchi, and Robert G. Hausmann. 2001. Learning from human tutoring. *Cognitive Science*, 25:471–533.

Michelene T.H. Chi. 1996. Constructing self-explanations and scaffolded explanations in tutoring. *Applied Cognitive Psychology*, 10:33–49.

Andrew Corrigan-Halpern. 2006. *Feedback in Complex Learning: Considering the Relationship Between Utility and Processing Demands*. Ph.D. thesis, University of Illinois at Chicago.

Barbara Di Eugenio, Davide Fossati, Dan Yu, Susan Haller, and Michael Glass. 2005. Aggregation improves learning: Experiments in natural language generation for intelligent tutoring systems. In *ACL05, Proceedings of the 42nd Meeting of the Association for Computational Linguistics*, Ann Arbor, MI.

Martha Evens and Joel Michael. 2006. *One-on-one Tutoring by Humans and Machines*. Mahwah, NJ: Lawrence Erlbaum Associates.

Trude Heift. 2001. Error-specific and individualized feedback in a web-based language tutoring system: Do they read it? *ReCALL Journal*, 13(2):129–142.

M. R. Lepper, M. Drake, and T. M. O'Donnell-Johnson. 1997. Scaffolding techniques of expert human tutors. In K. Hogan and M. Pressley, editors, *Scaffolding student learning: Instructional approaches and issues*, pages 108–144. Brookline Books, New York.

Bing Liu. 2006. *Web Data Mining*. Springer, Berlin.

Xin Lu. 2007. *Expert Tutoring and Natural Language Feedback in Intelligent Tutoring Systems*. Ph.D. thesis, University of Illinois at Chicago.

Antonija Mitrović, Pramuditha Suraweera, Brent Martin, and A. Weerasinghe. 2004. DB-suite: Experiences with three intelligent, web-based database tutors. *Journal of Interactive Learning Research*, 15(4):409–432.

Stellan Ohlsson, Barbara Di Eugenio, Bettina Chow, Davide Fossati, Xin Lu, and Trina C. Kershaw. 2007. Beyond the code-and-count analysis of tutoring dialogues. In *AIED07, 13th International Conference on Artificial Intelligence in Education*.

Stellan Ohlsson. 1996. Learning from performance errors. *Psychological Review*, 103:241–262.

N. K. Person, A. C. Graesser, L. Bautista, E. C. Mathews, and the Tutoring Research Group. 2001. Evaluating student learning gains in two versions of AutoTutor. In J. D. Moore, C. L. Redfield, and W. L. Johnson, editors, *Artificial intelligence in education: AI-ED in the wired and wireless future*, pages 286–293. Amsterdam: IOS Press.

Alexander Renkl. 2002. Learning from worked-out examples: Instructional explanations supplement self-explanations. *Learning and Instruction*, 12:529–556.

Kurt Van Lehn, Collin Lynch, Kay Schulze, Joel A. Shapiro, Robert H. Shelby, Linwood Taylor, Don J. Treacy, Anders Weinstein, and Mary C. Wintersgill. 2005. The Andes physics tutoring system: Five years of evaluations. In G. I. McCalla and C. K. Looi, editors, *Artificial Intelligence in Education Conference*. Amsterdam: IOS Press.

Melanie R. Weaver. 2006. Do students value feedback? Student perceptions of tutors' written responses. *Assessment and Evaluation in Higher Education*, 31(3):379–394.

Arabic Language Modeling with Finite State Transducers

Ilana Heintz
Department of Linguistics
The Ohio State University
Columbus, OH
heintz.38@osu.edu*

Abstract

In morphologically rich languages such as Arabic, the abundance of word forms resulting from increased morpheme combinations is significantly greater than for languages with fewer inflected forms (Kirchhoff et al., 2006). This exacerbates the out-of-vocabulary (OOV) problem. Test set words are more likely to be unknown, limiting the effectiveness of the model. The goal of this study is to use the regularities of Arabic inflectional morphology to reduce the OOV problem in that language. We hope that success in this task will result in a decrease in word error rate in Arabic automatic speech recognition.

1 Introduction

The task of language modeling is to predict the next word in a sequence of words (Jelinek et al., 1991). Predicting words that have not yet been seen is the main obstacle (Gale and Sampson, 1995), and is called the Out of Vocabulary (OOV) problem. In morphologically rich languages, the OOV problem is worsened by the increased number of morpheme combinations.

Berton et al. (1996) and Geutner (1995) approached this problem in German, finding that language models built on decomposed words reduce the OOV rate of a test set. In Carki et al. (2000), Turkish words are split into syllables for language modeling, also reducing the OOV rate (but not improving WER). Morphological decomposition is also used to boost language modeling scores in Korean (Kwon, 2000) and Finnish (Hirsimäki et al., 2006).

We approach the processing of Arabic morphology, both inflectional and derivational, with finite state machines (FSMs). We use a technique that produces many morphological analyses for each word, retaining information about possible stems, affixes, root letters, and templates. We build our language models on the morphemes generated by the analyses. The FSMs generate spurious analyses. That is, although a word out of context may have several morphological analyses, in context only one such analysis is correct. We retain all analyses. We expect that any incorrect morphemes that are generated will not affect the predictions of the model, because they will be rare, and the language model introduces bias towards frequent morphemes. Although many words in a test set may not have occurred in a training set, the morphemes that make up that word likely will have occurred. Using many decompositions to describe each word sets apart this study from other similar studies, including those by Wang and Vergyri (2006) and Xiang et al. (2006).

This study differs from previous research on Arabic language modeling and Arabic automatic speech recognition in two other ways. To promote cross-dialectal use of the techniques, we use properties of Arabic morphology that we assume to be common to many dialects. Also, we treat morphological analysis and vowel prediction with a single solution.

An overview of Arabic morphology is given in Section 2. A description of the finite state machine process used to decompose the Arabic words into

*This work was supported by a student-faculty fellowship from the AFRL/Dayton Area Graduate Studies Insititute, and worked on in partnership with Ray Slyh and Tim Anderson of the Air Force Research Labs.

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 37–42,
Columbus, June 2008.

morphemes follows in Section 3. The experimental language model training procedure and the procedures for training two baseline language models are discussed in Section 4. We evaluate all three models using average negative log probability and coverage statistics, discussed in Section 5.

2 Arabic Morphology

This section describes the morphological processes responsible for the proliferation of word forms in Arabic. The discussion is based on information from grammar textbooks such as that by Haywood and Nahmad (1965), as well as descriptions in various Arabic NLP articles, including that by Kirchhoff et al. (2006).

Word formation in Arabic takes place on two levels. Arabic is a root-and-pattern language in which many vocalic and consonantal patterns combine with semantic roots to create surface forms. A root, usually composed of three letters, may encode more than one meaning. Only by combining a root with a pattern does one create a meaningful and specific term. The combination of a root with a pattern is a *stem*. In some cases, a stem is a complete surface form; in other cases, affixes are added.

The second level of word formation is inflectional, and is usually a concatenative process. Inflectional affixes are used to encode person, number, gender, tense, and mood information on verbs, and gender, number, and case information on nouns. Affixes are a closed class of morphemes, and they encode predictable information. In addition to inflection, cliticization is common in Arabic text. Prepositions, conjunctions, and possessive pronouns are expressed as clitics.

This combination of templatic derivational morphology and concatenative inflectional morphology, together with cliticization, results in a rich variation in word forms. This richness is in contrast with the slower growth in number of English word forms. As shown in Table 1, the Arabic stem /drs/, meaning *to study*, combines with the present tense verb pattern "*CC*u*C*u", where the 'C' represents a root letter, to form the present tense stem *drusu*. This stem can be combined with 11 different combinations of inflectional affixes, creating as many unique word forms.

Table 1 can be expanded with stems from the same root representing different tenses. For instance, the stem *daras* means *studied*. Or, we can combine the root with a different pattern to obtain different meanings, for instance, to teach or to learn. Each of these stems can combine with the same or different affixes to create additional word forms.

Transliteration	Translation	Affixes
adrusu	I study	a-
nadrusu	we study	na-
tadrusu	you (ms) study	ta-
tadrusina	you (fs) study	ta- ,-ina
tadrusAn	you (dual) study	ta-, -An
tadrusun	you (mp) study	ya-, -n
tadrusna	you (fp) study	ta-, -na
yadrusu	he studies	ya-
tadrusu	she studies	ta-
yadrusan	they (dual) study	ya-, -An
yadrusun	they (mp) study	ya-, -n
yadrusna	they (fp) study	ya-, -na

Table 1: An Example of Arabic Inflectional Morphology

Adding a single clitic to the words in Table 1 will double the number of forms. For instance, the word *adrusu*, meaning *I study*, can take the enclitic 'ha', to express *I study it*. Some clitics can be combined, increasing again the number of possible word forms.

Stems differ in some ways that do not surface in the Arabic orthography. For instance, the pattern "*CC*i*C*u" differs from "*CC*u*C*u" only in one short vowel, which is encoded orthographically as a frequently omitted diacritic. Thus, *adrisu* and *adrusu* are homographs, but not homophones. This property helps decrease the number of word forms, but it causes ambiguity in morphological analyses. Recovering the quality of short vowels is a significant challenge in Arabic natural language processing.

This abundance of unique word forms in Modern Standard Arabic is problematic for natural language processing (NLP). NLP tasks usually require that some analysis be provided for each word (or other linguistic unit) in a given data set. For instance, in spoken word recognition, the decoding process makes use of a language model to predict the words that best fit the acoustic signal. Only words that have been seen in the language model's training data will be proposed. Because of the immense number of possible word forms in Arabic, it is highly proba-

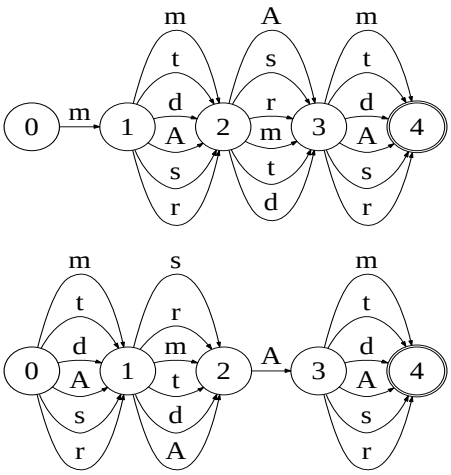


Figure 1: Two templates, m*CCC* and *CC*A*C* as finite state recognizers, with a small sample alphabet of letters A, d, m, r, s, and t.

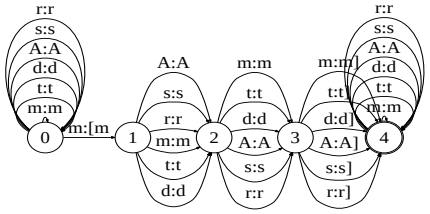


Figure 2: The first template above, now a transducer, with affixes accepted, and the stem separated by brackets in the output.

ble that the words in an acoustic signal will not have been present in the language model's training text, and incorrect words will be predicted. We use information about the morphology of Arabic to create a more flexible language model. This model should encounter fewer unseen forms, as the units we use to model the language are the more frequent and predictable morphemes, as opposed to full word forms. As a result, the word error rate is expected to decrease.

3 FSM Analyses

This section describes how we derive, for each word, a lattice that describes all possible morphological decompositions for that word. We start with a group of templates that define the root consonant positions, long vowels, and consonants for all Arabic regular and augmented stems. For instance, where *C* represents a root consonant, three possible templates are

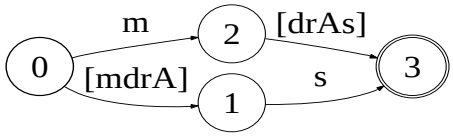


Figure 3: Two analyses of the word "mdrAs", as produced by composing a word FSM with the template FSMs above.

CCC, m*CCC*, and *CA*C*C*. We build a finite state recognizer for each of the templates, and in each case, the *C* arcs are expanded, so that every possible root consonant in the vocabulary has an arc at that position. The two examples in Figure 1 show the patterns m*CCC* and *CC*A*C* and a short sample alphabet.

At the start and end node of each template recognizer, we add arcs with self-loops. This allows any sequence of consonants as an affix. To track stem boundaries, we add an open bracket to the first stem arc, and a close bracket to the final stem arc. The templates are compiled into finite state transducers. Figure 2 shows the result of these additions.

For each word in the vocabulary, we define a simple, one-arc-per-letter finite state recognizer. We compose this with each of the templates. Some number of analyses result from each composition. That is, a single template may not compose with the word, may compose with it in a unique way, or may compose with the word in several ways. Each of the successful compositions produces a finite state recognizer with brackets surrounding the stem. We use a script to collapse the arcs within the stem to a single arc. The result is shown in Figure 3, where the word "mdrAs" has two analyses corresponding to the two templates shown. We store a lattice as in Figure 3 for each word.

The patterns that we use to constrain the stem forms are drawn from Haywood and Nahmad (1965). These patterns also specify the short vowel patterns that are used with words derived from each pattern. An option is to simply add these short vowels to the output symbols in the template FSTs. However, because several short vowel options may exist for each template, this would greatly increase the size of the resulting lattices. We postpone this effort. In this work, we focus solely on the usefulness of the unvoweled morphological decompositions.

We do not assess or need to assess the accuracy of

the morphological decompositions. Our hypothesis is that by having many possible decompositions per word, the frequencies of various affixes and stems across all words will lead the model to the strongest predictions. Even if the final predictions are not prescriptively correct, they may be the most useful decompositions for the purpose of speech decoding.

4 Procedure

We compare a language model built on multiple segmentations as determined by the FSMs described above to two baseline models. We call our experimental model FSM-LM; the baseline models use word-based n-grams (WORD), and pre-defined affix segmentations (AFFIX). Our data set in this study is the TDT4 Arabic broadcast news transcriptions (Kong and Graff, 2005). Because of time and memory constraints, we built and evaluated all models on only a subsection of the training data, 100 files of TDT4, balanced across the years of collection, and containing files from each of the 4 news sources. We use 90 files for training, comprising about 6.3 million unvoweled word tokens, and 10 files for testing, comprising about 700K word tokens, and around 5K sentences. The size of the vocabulary is 104757. We use ten-fold cross-validation in our evaluations.

4.1 Experimental Model

We extract the vocabulary of the training data, and compile the word lattices as described in Section 3. The union of all decompositions (a lattice) for each individual word is stored separately.

For each sentence of training data, we concatenate the lattices representing each word in that sentence. We use SRILM (Stolcke, 2002) to calculate the posterior expected n-gram count for morpheme sequences up to 4-grams in the sentence-long lattice. The estimated frequency of an n-gram N is calculated as the number of occurrences of that n-gram in the lattice, divided by the number of paths in the lattice. This is true so long as the paths are equally weighted; at this point in our study, this is the case.

We merge the n-gram counts over all sentences in all of the training files. Next, we estimate a language model based on the n-gram counts, using only the 64000 most frequent morphemes, since we expect this vocabulary size may be a limitation of our ASR system. Also, by limiting the vocabulary size of all of our models (including the baseline models described below), we can make a fairer comparison among the models. We use Good-Turing smoothing to account for unseen morphemes, all of which are replaced with a single "unknown" symbol.

In later work, we will apply our LM statistics to the lattices, and recalculate the path weights and estimated counts. In this study, the paths remain equally weighted.

We evaluate this model, which we call FSM-LM, with respect to two baseline models.

4.2 Baseline Models

For the WORD model, we do no manipulation to the training or test sets beyond the normalization that occurs as a preprocessing step (hamza normalization, replacement of problematic characters). We build a word-based 4-gram language model using the 64000 most frequent words and Good-Turing smoothing.

For the AFFIX model, we first define the character strings that are considered affixes. We use the same list of affixes as in Xiang et al. (2006), which includes 12 prefixes and 34 suffixes. We add to the lists all combinations of two prefixes and two suffixes. We extract the vocabulary from the training data, and for each word, propose a single segmentation, based on the following constraints:

1. If the word has an acceptable prefix-stem-suffix decomposition, such that the stem is at least 3 characters long, choose it as the correct decomposition.
2. If only one affix is found, make sure the remainder is at least 3 characters long, and is not also a possible affix.
3. If the word has prefix-stem and stem-suffix decompositions, use the longest affix.
4. If the longest prefix and longest suffix are equal length, choose the prefix-stem decomposition.

We build a dictionary that relates each word to a single segmentation (or no segmentation). We segment the training and test texts by replacing each word with its segmentation. Morphemes are separated by whitespace. The language model is built by counting 4-grams over the training data, then using only the most frequent 64000 morphemes in estimating a language model with Good-Turing smoothing.

	WORD	AFFIX	FSM-LM
Avg Neg Log Prob	4.65	5.30	4.56
Coverage (%):			
Unigram	96.03	99.30	98.89
Bigram	17.81	53.13	69.56
Trigram	1.52	11.89	27.25
Four-gram	.37	3.42	9.62

Table 2: Average negative log probability and coverage results for one experimental language model (FSM-LM) and two baseline language models. Results are averages over 10 folds.

5 Evaluation

For each model, the test set undergoes the same manipulation as the train set; words are left alone for the WORD model, split into a single segmentation each for the AFFIX model, or their FSM decompositions are concatenated.

Language models are often compared using the perplexity statistic:

$$PP(x_1 \dots x_n) = 2^{-\frac{1}{n}\sum_{x_i=4}^{n} logP(x_i|x_{i-1}^{i-3})} \quad (1)$$

Perplexity represents the average branching factor of a model; that is, at each point in the test set, we calculate the entropy of the model. Therefore, a lower perplexity is desired.

In the AFFIX and FSM-LM models, each word is split into several parts. Therefore, the value $\frac{1}{n}$ would be approximately three times smaller for these models, giving them an advantage. To make a more even comparison, we calculate the geometric mean of the n-gram transition probabilities, dividing by the number of *words* in the test set, not morphemes, as in Kirchhoff et al. (2006). The log of this equation is:

$$AvgNegLogProb(x_1 \dots x_n) = -\frac{1}{N}\sum_{i=4}^{n} logP(x_i|x_{i-1}^{i-3}) \quad (2)$$

where n is the number of morphemes or words in the test set, depending on the model, and N is the number of *words* in the test set, and $log\ P(x_i|x_{i-1}^{i-3})$ is the log probability of the item x_i given the 3-item history (calculated in base 10, as this is how the SRILM Toolkit is implemented). Again, we are looking for a low score.

In the FSM-LM, each test sentence is represented by a lattice of paths. To determine the negative log probability of the sentence, we score all paths of the sentence according to the equations above, and record the maximum probability. This reflects the likely procedure we would use in implementing this model within an ASR task.

We see in Table 2 that the average negative log probability of the FSM-LM is lower than that of either the WORD or AFFIX model. The average across 10 folds reflects the pattern of scores for each fold. We conclude from this that the FSM model of predicting morphemes is more effective than - or more conservatively, at least as effective as - a static decomposition, as in the AFFIX model. Furthermore, we have successfully reproduced the results of Xiang et al. (2006) and Kirchhoff et al. (2006), among others, that modeling Arabic with morphemes is more effective than modeling with whole word forms.

We also calculate the coverage of each model: the percentage of units in the test set that are given probabilities in the language model. For the FSM model, only the morphemes in the best path are counted. The coverage results are reported in Table 2 as the average coverage over the 10 folds. Both the AFFIX and FSM-LM models showed improved coverage as compared to the WORD model, as expected. This means that we reduce the OOV problem by using morphemes instead of whole words. The AFFIX model has the best coverage of unigrams because only new stems, not new affixes, are proposed in the test set. That is, the same fixed set of affixes are used to decompose the test set as the train set, however, unseem stems may appear. In the FSM-LM, there are no restrictions on the affixes, therefore, unseen affixes may appear in the test set, as well as new stems, lowering the unigram coverage of the test set. For larger n-grams, however, the FSM-LM model has the best coverage. This is due to keeping all decompositions until test time, then allowing the language model to define the most likely sequences, rather than specifying a single decomposition for each word.

A 4-gram of words will tend to cover more context than a 4-gram of morphemes; therefore, the word 4-grams will exhibit more sparsity than the morpheme 4-grams. We compare, for a single train-

	WORD	AFFIX	FSM-LM
unigrams	4.97	5.84	5.60
bigrams	4.95	5.70	4.61
trigrams	4.95	5.69	4.56
four-grams	4.95	5.69	4.57

Table 3: Comparison of n-gram orders across language model types.

test fold, how lower order n-grams compare among the models. The results are shown in Table 3. We find that for lower-order n-grams, the word model performs best. As the n-grams get larger, the sparsity problem favors the FSM-LM, which has the best overall score of all models shown. Apparently, the frequencies of 3- and 4-grams are not big enough to make a big difference in the evaluation. This is likely due to the small size of our corpus, and we expect the result would change if we were to use all of the TDT4 corpus, rather than a 100 file portion of the corpus.

6 Conclusion & Future Work

It has been shown that reduced perplexity scores do not necessarily correlate with reduced word error rates in an ASR task (Berton et al., 1996). This is because the perplexity (or in this case, average negative log probability) statistic does not take into account the acoustic confusability of the items being considered. However, the average negative log probability score is a useful tool as a proof-of-concept, giving us reason to believe that we may be successful in implementing this model within an ASR task.

The real test of this model is its ability to predict short vowels. The average negative log probability scores may lead us to believe that the FSM-LM is only marginally better than the WORD or AFFIX model, and the differences may not be apparent in an ASR application. However, only the FSM-LM model allows for the opportunity to predict short vowels, by arranging the FSMs as finite state transducers with short vowel information encoded as part of the stem patterns.

We will continue to tune the language model by applying the language model weights to the decomposition paths and re-estimating the language model. Also, we will expand the language model to include more training data. We will implement the model within an Arabic ASR system, with and without short vowel hypotheses. Furthermore, we are interested to see how well the application of these templates and this framework will apply to other Arabic dialects.

References

A Berton, P Fetter, and P Regel-Brietzmann. 1996. Compound words in large vocabulary German speech recognition system. In *Proceedings of ICSLP 96*, pages 1165–1168.

K. Carki, P. Geutner, and T. Schultz. 2000. Turkish lvcsr: Towards better speech recognition for agglutinative languages. In *ICASSP 2000*, pages 134–137.

William A. Gale and Geoffrey Sampson. 1995. Good-Turing frequency estimation without tears. *Journal of Quantitative Linguistics*, 22:217–37.

P Geutner. 1995. Using morphology towards better large-vocabulary speech recognition systems. In *Proceedings of ICASSP-95*, volume 1, pages 445–448.

J.A. Haywood and H.M. Nahmad. 1965. *A New Arabic Grammar of the Written Language*. Lund Humphries, Burlington, VT.

Teemu Hirsimäki, Mathias Creutz, Vesa Siivola, Mikko Kurimo, Sami Virpioja, and Janne Pylkkönen. 2006. Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech and Language*, 20:515–541.

F. Jelinek, B. Merialdo, S. Roukos, and M. Strauss. 1991. A dynamic language model for speech recognition. In *Proc. Wkshp on Speech and Natural Language*, pages 293–295, Pacific Grove, California. ACL.

Katrin Kirchhoff, Dimitra Vergyri, Kevin Duh, Jeff Bilmes, and Andreas Stolcke. 2006. Morphology-based language modeling for conversational Arabic speech recognition. *Computer Speech and Language*, 20(4):589–608.

Junbo Kong and David Graff. 2005. TDT4 multilingual broadcast news speech corpus.

Oh-Wook Kwon. 2000. Performance of LVCSR with morpheme-based and syllable-based recognition units. In *Proceedings of ICASSP '00*, volume 3, pages 1567–1570.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proc. Intl. Conf. Spoken Language Processing*, Denver, Colorado.

Wen Wang and Dimitra Vergyri. 2006. The use of word n-grams and parts of speech for hierarchical cluster language modeling. In *Proceedings of ICASSP 2006*, pages 1057–1060.

Bing Xiang, Kham Nguyen, Long Nguyen, Richard Schwartz, and John Makhoul. 2006. Morphological decomposition for Arabic broadcast news transcription. In *Proc. ICASSP 2006*, pages 1089–1092.

Impact of Initiative on Collaborative Problem Solving *

Cynthia Kersey
Department of Computer Science
University of Illinois at Chicago
Chicago, Illinois 60613
ckerse2@uic.edu

Abstract

Even though collaboration in peer learning has been shown to have a positive impact for students, there has been little research into collaborative peer learning dialogues. We analyze such dialogues in order to derive a model of knowledge co-construction that incorporates initiative and the balance of initiative. This model will be embedded in an artificial agent that will collaborate with students.

1 Introduction

While collaboration in dialogue has long been researched in computational linguistics (Chu-Carroll and Carberry, 1998; Constantino-González and Suthers, 2000; Jordan and Di Eugenio, 1997; Lochbaum and Sidner, 1990; Soller, 2004; Vizcaíno, 2005), there has been little research on collaboration in peer learning. However, this is an important area of study because collaboration has been shown to promote learning, potentially for all of the participants (Tin, 2003). Additionally, while there has been a focus on using natural language for intelligent tutoring systems (Evens et al., 1997; Graesser et al., 2004; VanLehn et al., 2002), peer to peer interactions are notably different from those of expert-novice pairings, especially with respect to the richness of the problem-solving deliberations and negotiations. Using natural language in collaborative learning could have a profound impact on the way in which educational applications engage students in learning.

*This work is funded by NSF grants 0536968 and 0536959.

There are various theories as to why collaboration in peer learning is effective, but one of that is commonly referenced is co-construction (Hausmann et al., 2004). This theory is a derivative of constructivism which proposes that students construct an understanding of a topic by interpreting new material in the context of prior knowledge (Chi et al., 2001). Essentially, students who are active in the learning process are more successful. In a collaborative situation this suggests that all collaborators should be active participants in order to have a successful learning experience. Given the lack of research in modeling peer learning dialogues, there has been little study of what features of dialogue characterize co-construction. I hypothesize that since instances of co-construction closely resemble the concepts of control and initiative, these dialogue features can be used as identifiers of co-construction.

While there is some dispute as to the definitions of control and initiative (Jordan and Di Eugenio, 1997; Chu-Carroll and Brown, 1998), it is generally accepted that one or more threads of control pass between participants in a dialogue. Intuitively, this suggests that tracking the transfer of control can be useful in determining when co-construction is occurring. Frequent transfer of control between participants would indicate that they are working together to solve the problem and perhaps also to construct knowledge.

The ultimate goal of this research is to develop a model of co-construction that incorporates initiative and the balance of initiative. This model will be embedded in KSC-PaL, a natural language based peer agent that will collaborate with students to solve

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 43–48,
Columbus, June 2008.

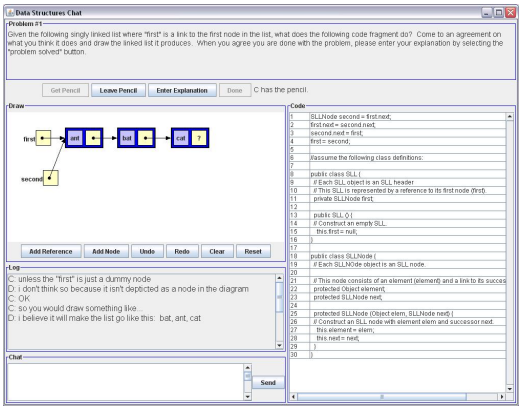


Figure 1: The data collection interface

problems in the domain of computer science data structures.

In section 2, I will describe how we collected the dialogues and the initial analysis of those dialogues. Section 3 details the on-going annotation of the corpus. Section 4 describes the future development of the computational model and artificial agent. This is followed by the conclusion in section 5.

2 Data Collection

In a current research project on peer learning, we have collected computer-mediated dialogues between pairs of students solving program comprehension and error diagnosis problems in the domain of data structures. The data structures that we are focusing on are (1) linked lists, (2) stacks and (3) binary search trees. This domain was chosen because data structures and their related algorithms are one of the core components of computer science education and a deep understanding of these topics is essential to a strong computer science foundation.

2.1 Interface

A computer mediated environment was chosen to more closely mimic the situation a student will have to face when interacting with KSC-PaL, the artificial peer agent. After observing face-to-face interactions of students solving these problems, I developed an interface consisting of four distinct areas (see Figure 1):

1. Problem display: Displays the problem description that is retrieved from a database.
2. Code display: Displays the code from the problem statement. The students are able to make changes to the code, such as crossing-out lines and inserting lines, as well as undoing these corrections.
3. Chat Area: Allows for user input and an interleaved dialogue history of both students participating in the problem solving. The history is logged for analysis.
4. Drawing area: Here users can diagram data structures to aid in the explanation of parts of the problem being solved. The drawing area has objects representing nodes and links. These objects can then be placed in the drawing area to build lists, stacks or trees depending on the type of problem being solved.

The changes made in the shared workspace (drawing and code areas) are logged and propagated to the partner's window. In order to prevent users from making changes at the same time, I implemented a system that allows only one user to draw or make changes to code at any point in time. In order to make a change in the shared workspace, a user must request the "pencil" (Constantino-González and Suthers, 2000). If the pencil is not currently allocated to her partner, the user receives the pencil and can make changes in the workspace. Otherwise, the partner is informed, through both text and an audible alert, that his peer is requesting the pencil. The chat area, however, allows users to type at the same time, although they are notified by a red circle at the top of the screen when their partner is typing. While, this potentially results in interleaved conversations, it allows for more natural communication between the peers.

Using this interface, we collected dialogues for a total of 15 pairs where each pair was presented with five problems. Prior to the collaborative problem solving activities, the participants were individually given pre-tests and at the conclusion of the session, they were each given another test, the post-test. During problem solving the participants were seated in front of computers in separate rooms and all problem solving activity was conducted using the computer-mediated interface. The initial exercise let the users become acquainted with the interface. The

Predictor	Prob. 3 (Lists)	Prob. 4 (Stacks)	Prob. 5 (Trees)
Pre-Test	0.530 (p=0.005)	0.657 (p=0.000)	0.663 (p=0.000)
Words	0.189 (p=0.021)		
Words per Turn	0.141 (p=0.049)		
Pencil Time	0.154 (p=0.039)		
Total Turns	0.108 (p=0.088)		
Code Turns			0.136 (p=0.076)

Table 1: Post-test Score Predictors (R^2)

participants were allowed to ask questions regarding the interface and were limited to 30 minutes to solve the problem. The remaining exercises had no time limits, however the total session, including pre-test and post-test could not exceed three hours. Therefore not all pairs completed all five problems.

2.2 Initial Analysis

After the completion of data collection, I established that the interface and task were conducive to learning by conducting a paired t-test on the pre-test and post-test scores. This analysis showed that the post-test score was moderately higher than the pre-test score (t(30)=2.83; p=0.007; effect size = 0.3).

I then performed an initial analysis of the collected dialogues using linear regression analysis to identify correlations between actions of the dyads and their success at solving the problems presented to them. Besides the post-test, students solutions to the problems were scored, as well; this is what we refer to as problem solving success. The participant actions were also correlated with post-test scores and learning gains (the difference between post-test score and pre-test score). The data that was analyzed came from three of the five problems for all 15 dyads, although not all dyads attempted all three problems. Thus, I analyzed a total of 40 subdialogues. The problems that were analyzed are all error diagnosis problems, but each problem involves a different data structure - linked list, array-based stack and binary search tree. Additionally, I analyzed the relationship between initiative and post-test score, learning gain and successful problem solving. Before embarking on an exhaustive manual annotation of initiative, I chose to get a sense of whether initiative may indeed affect learning in this context by automatically tagging for initiative using an approximation of Walker and Whittaker's utterance based allocation of control rules (Walker and Whittaker, 1990). In this scheme, first each turn in the dialogue must be tagged as either: (1) an assertion, (2) a command, (3) a question or (4) a prompt (turns not expressing propositional content). This was done automatically, by marking turns that end in a question mark as questions, those that start with a verb as commands, prompts from a list of commonly used prompts (e.g. ok, yeah) and the remaining turns as assertions. Control is then allocated by using the following rules based on the turn type:

1. Assertion: Control is allocated to the speaker unless it is a response to a question.

2. Command: Control is allocated to the speaker.

3. Question: Control is allocated to the speaker, unless it is a response to a question or a command.

4. Prompt: Control is allocated to the hearer.

Since the dialogues also have a graphics component, all drawing and code change moves had control assigned to the peer drawing or making the code change.

The results of the regression analysis are summarized in tables 1 and 2, with blank cells representing non-significant correlations. Pre-test score, which represents the student's initial knowledge and/or aptitude in the area, was selected as a feature because it is important to understand the strength of the correlation between previous knowledge and post test score when identifying additional correlating features (Yap, 1979). The same holds for the time related features (pencil time and total time). The remaining correlations and trends to correlation suggest that participation is an important factor in successful collaboration. Since a student is more likely to take initiative when actively participating in prob-

Predictor	Prob. 3 (Lists)	Prob. 4 (Stacks)	Prob. 5 (Trees)
Pre-Test	0.334 (p=0.001)	0.214 (p=0.017)	0.269 (p=0.009)
Total Time	0.186 (p=0.022)	0.125 (p=0.076)	0.129 (p=0.085)
Total Turns	0.129 (p=0.061)	0.134 (p=0.065)	
Draw Turns	0.116 (p=0.076)	0.122 (p=0.080)	
Code Turns		0.130 (p=0.071)	

Table 2: Problem Score Predictors (R^2)

lem solving, potentially there there is a relation between these participation correlations and initiative.

An analysis of initiative shows that there is a correlation of initiative and successful collaboration. In problem 3, learning gain positively correlates with the number of turns where a student has initiative ($R^2 = 0.156, p = 0.037$). And in problem 4, taking initiative through drawing has a positive impact on post-test score ($R^2 = 0.155, p = 0.047$).

3 Annotation

Since the preliminary analysis showed a correlation of initiative with learning gain, I chose to begin a thorough data analysis by annotating the dialogues with initiative shifts. Walker and Whittaker claim that initiative encompasses both dialogue control and task control (Walker and Whittaker, 1990), however, several others disagree. Jordan and Di Eugenio propose that control and initiative are two separate features in collaborative problem solving dialogues (Jordan and Di Eugenio, 1997). While control and initiative might be synonymous for the dialogues analyzed by Walker and Whittaker where a master-slave assumption holds, it is not the case in collaborative dialogues where no such assumption exists. Jordan and Di Eugenio argue that the notion of control should apply to the dialogue level, while initiative should pertain to the problem-solving goals. In a similar vein, Chu-Carroll and Brown also argue for a distinction between control and initiative, which they term task initiative and dialogue initiative (Chu-Carroll and Brown, 1998). Since there is no universally agreed upon definition for initiative, I have decided to annotate for both dialogue initiative and task initiative. For dialogue initiative annotation, I am using Walker and Whittaker's utterance based allocation of control rules (Walker and Whittaker, 1990), which are widely used to identify dialogue initiative. For task initiative, I have derived an annotation scheme based on other research in the area. According to Jordan and Di Eugenio, in problem solving (task) initiative the agent takes it upon himself to address domain goals by either (1)proposing a solution or (2)reformulating goals. In a similar vein, Guinn (Guinn, 1998) defines task initiative as belonging to the participant who dictates which decomposition of the goal will be used by both participants during problem-solving. A third definition is from Chu-Carroll and Brown. They suggest that task initiative tracks the lead in development of the agent's plan. Since the primary goal of the dialogues studied by Chu-Carroll and Brown is to develop a plan, this could be re-worded to state that task initiative tracks the lead in development of the agent's goal. Combining these definitions, task initiative can be defined as *any action by a participant to either achieve a goal directly, decompose a goal or reformulate a goal*. Since the goals of our problems are understanding and potentially correcting a program, actions in our domain that show task initiative include actions such as explaining what a section of code does or identifying a section of code that is incorrect.

Two coders, the author and an outside annotator, have coded 24 dialogues (1449 utterances) for both dialogue and task initiative. This is approximately 45% of the corpus. The resulting intercoder reliability, measured with the Kappa statistic, is 0.77 for dialogue initiative annotation and 0.68 for task initiative, both of which are high enough to support tentative conclusions. Using multiple linear regression analysis on these annotated dialogues, I found that, in a subset of the problems, there was a signicant correlation between post-test score (after removing the effects of pre-test scores) and the number of switches in dialogue initiative (R^2 =0.157, p=0.014). Also, in the same subset, there was a correlation between post-test score and the number of turns that a student had initiative (R^2 =0.077, p=0.065). This suggests that both taking the ini-

tiative and taking turns in leading problem solving results in learning.

Given my hypothesis that initiative can be used to identify co-construction, the next step is to annotate the dialogues using a subset of the DAMSL scheme (Core and Allen, 1997) to identify episodes of co-construction. Once annotated, I will use machine learning techniques to identify co-construction using initiative as a feature. Since this is a classification problem, algorithms such as Classification Based on Associations (Liu, 2007) will be used. Additionally, I will explore those algorithms that take into account the sequence of actions, such as hidden Markov models or neural networks.

4 Computational Model

The model will be implemented as an artificial agent, KSC-PaL, that interacts with a peer in collaborative problem solving using an interface similar to the one that was used in data collection (see Figure 1). This agent will be an extension of the TuTalk system, which is designed to support natural language dialogues for educational applications (Jordan et al., 2006). TuTalk contains a core set of dialogue system modules that can be replaced or enhanced as required by the application. The core modules are understanding and generation, a dialogue manager which is loosely characterized as a finite state machine with a stack and a student model. To implement the peer agent, I will replace TuTalk's student model and add a planner module.

Managing the information state of the dialogue (Larsson and Traum, 2000), which includes the beliefs and intentions of the participants, is important in the implementation of any dialogue agent. KSC-PaL will use a student model to assist in management of the information state. This student model tracks the current state of problem solving as well as estimates the student's knowledge of concepts involved in solving the problem by incorporating problem solution graphs (Conati et al., 2002). Solution graphs are Bayesian networks where each node represents either an action required to solve the problem or a concept required as part of problem solving. After analyzing our dialogues, I realized that the solutions to the problems in our domain are different from standard problem-solving tasks. Given that our tasks are program comprehension tasks and that the dialogues are peer led, there can be no assumption as to the order in which a student will analyze code statements. Therefore a graph comprised of connected subgraphs that each represent a section of the code more closely matches what I observed in our dialogues. So, we are using a modified version of solution graphs that has clusters of nodes representing facts that are relevant to the problem. Each cluster contains facts that are dependent on one another. For example, one cluster represents facts related to the push method for a stack. As the code is written, it would be impossible to comprehend the method without understanding the prefix notation for incrementing. A user's utterances and actions can then be matched to the nodes within the clusters. This provides the agent with information related to the student's knowledge as well as the current topic under discussion.

A planner module will be added to TuTalk to provide KSC-PaL with a more sophisticated method of selecting scripts. Unlike TuTalk's dialogue manager which uses a simple matching of utterances to concepts in order to determine the script to be followed, KSC-PaL's planner will incorporate the results of the data analysis above and will also include the status of the student's knowledge, as reflected in the student model, in making script selections. This planner will potentially be a probabilistic planner such as the one in (Lu, 2007).

5 Conclusion

In conclusion, we are developing a computational model of knowledge construction which incorporates initiative and the balance of initiative. This model will be embedded in an artificial agent that collaborates with students to solve data structure problems. As knowledge construction has been shown to promote learning, this research could have a profound impact on educational applications by changing the way in which they engage students in learning.

Acknowledgments

The graphical interface is based on a graphical interface developed by Davide Fossati for an intelligent tutoring system in the same domain.

References

Michelene T. H. Chi, Stephanie A. Siler, Jeong Heisawn, Takashi Yamauchi, and Robert G. Hausmann. 2001. Learning from human tutoring. *Cognitive Science*, 25(4):471–533.

Jennifer Chu-Carroll and Michael K. Brown. 1998. An evidential model for tracking initiative in collaborative dialogue interactions. *User Modeling and User-Adapted Interaction*, 8(3–4):215–253, September.

Jennifer Chu-Carroll and Sandra Carberry. 1998. Collaborative response generation in planning dialogues. *Computational Linguistics*, 24(3):355–400.

Cristina Conati, Abigail Gertner, and Kurt Vanlehn. 2002. Using bayesian networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interaction*, 12(4):371–417.

María de los Angeles Constantino-González and Daniel D. Suthers. 2000. A coached collaborative learning environment for entity-relationship modeling. *Intelligent Tutoring Systems*, pages 324–333.

Mark G. Core and James F. Allen. 1997. Coding dialogues with the DAMSL annotation scheme. In David Traum, editor, *Working Notes: AAAI Fall Symposium on Communicative Action in Humans and Machines*, pages 28–35, Menlo Park, California. American Association for Artificial Intelligence.

Martha W. Evens, Ru-Charn Chang, Yoon Hee Lee, Leem Seop Shim, Chong Woo Woo, Yuemei Zhang, Joel A. Michael, and Allen A. Rovick. 1997. Circsim-tutor: an intelligent tutoring system using natural language dialogue. In *Proceedings of the fifth conference on Applied natural language processing*, pages 13–14, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Arthur C. Graesser, Shulan Lu, George Tanner Jackson, Heather Hite Mitchell, Mathew Ventura, Andrew Olney, and Max M. Louwerse. 2004. Autotutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers*, 36:180–192(13), May.

Curry I. Guinn. 1998. An analysis of initiative selection in collaborative task-oriented discourse. *User Modeling and User-Adapted Interaction*, 8(3-4):255–314.

Robert G.M. Hausmann, Michelee T.H. Chi, and Marguerite Roy. 2004. Learning from collaborative problem solving: An analysis of three hypothesized mechanisms. In K.D Forbus, D. Gentner, and T. Regier, editors, *26th Annual Converence of the Cognitive Science Society*, pages 547–552, Mahwah, NJ.

Pamela W. Jordan and Barbara Di Eugenio. 1997. Control and initiative in collaborative problem solving dialogues. In *Working Notes of the AAAI Spring Symposium on Computational Models for Mixed Initiative*, pages 81–84, Menlo Park, CA.

Pamela W. Jordan, Michael Ringenberg, and Brian Hall. 2006. Rapidly developing dialogue systems that support learning studies. In *Proceedings of ITS06 Workshop on Teaching with Robots, Agents, and NLP*, pages 1–8.

Staffan Larsson and David R. Traum. 2000. Information state and dialogue management in the trindi dialogue move engine toolkit. *Nat. Lang. Eng.*, 6(3-4):323–340.

Bing Liu. 2007. *Web data mining: exploring hyperlinks, contents, and usage data.* Springer.

Karen E. Lochbaum and Candice L. Sidner. 1990. Models of plans to support communication: An initial report. In Thomas Dietterich and William Swartout, editors, *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 485–490, Menlo Park, California. AAAI Press.

Xin Lu. 2007. *Expert Tutoring and Natural Language Feedback in Intelligent Tutoring Systems*. Ph.D. thesis, University of Illinois at Chicago.

Amy Soller. 2004. Computational modeling and analysis of knowledge sharing in collaborative distance learning. *User Modeling and User-Adapted Interaction*, Volume 14(4):351–381, January.

Tan Bee Tin. 2003. Does talking with peers help learning? the role of expertise and talk in convergent group discussion tasks. *Journal of English for Academic Purposes*, 2(1):53–66.

Kurt VanLehn, Pamela W. Jordan, Carolyn Penstein Rosé, Dumisizwe Bhembe, Michael Böttner, Andy Gaydos, Maxim Makatchev, Umarani Pappuswamy, Michael A. Ringenberg, Antonio Roque, Stephanie Siler, and Ramesh Srivastava. 2002. The architecture of why2-atlas: A coach for qualitative physics essay writing. In *ITS '02: Proceedings of the 6th International Conference on Intelligent Tutoring Systems*, pages 158–167, London, UK. Springer-Verlag.

Aurora Vizcaíno. 2005. A simulated student can improve collaborative learning. *International Journal of Artificial Intelligence in Education*, 15(1):3–40.

Marilyn Walker and Steve Whittaker. 1990. Mixed initiative in dialogue: an investigation into discourse segmentation. In *Proceedings of the 28th annual meeting on Association for Computational Linguistics*, pages 70–78, Morristown, NJ, USA. Association for Computational Linguistics.

Kim Onn Yap. 1979. Pretest-posttest correlation and regression models. Presented at the Annual Meeting of the American Educational Research Association (63rd, San Francisco, California), April 8–12.

An Unsupervised Vector Approach to Biomedical Term Disambiguation: Integrating UMLS and Medline

Bridget T. McInnes
Computer Science Department
University of Minnesota Twin Cities
Minneapolis, MN 55155, USA
bthomson@cs.umn.edu

Abstract

This paper introduces an unsupervised vector approach to disambiguate words in biomedical text that can be applied to all-word disambiguation. We explore using contextual information from the Unified Medical Language System (UMLS) to describe the possible senses of a word. We experiment with automatically creating individualized stoplists to help reduce the noise in our dataset. We compare our results to SenseClusters and Humphrey et al. (2006) using the NLM-WSD dataset and with SenseClusters using conflated data from the 2005 Medline Baseline.

1 Introduction

Some words have multiple senses. For example, the word $cold$ could refer to a viral infection or the temperature. As humans, we find it easy to determine the appropriate sense (concept) given the context in which the word is used. For a computer, though, this is a difficult problem which negatively impacts the accuracy of biomedical applications such as medical coding and indexing. The goal of our research is to explore using information from biomedical knowledge sources such as the Unified Medical Language System (UMLS) and Medline to help distinguish between different possible concepts of a word.

In the UMLS, concepts associated with words and terms are enumerated via Concept Unique Identifiers (CUIs). For example, two possible senses of $cold$ are "C0009264: Cold Temperature" and "C0009443: Common Cold" in the UMLS release 2008AA. The UMLS is also encoded with different semantic and syntactic structures. Some such information includes related concepts and semantic types. A semantic type (ST) is a broad subject categorization assigned to a CUI. For example, the ST of "C0009264: Cold Temperature" is "Idea or Concept" while the ST for "C0009443: Common Cold" is "Disease or Syndrome". Currently, there exists approximately 1.5 million CUIs and 135 STs in the UMLS. Medline is an online database that contains 11 million references biomedical articles.

In this paper, we introduce an unsupervised vector approach to disambiguate words in biomedical text using contextual information from the UMLS and Medline. We compare our approach to Humphrey et al. (2006) and SenseClusters. The ability to make disambiguation decisions for words that have the same ST differentiates SenseClusters and our approach from Humphrey et al.'s (2006). For example, the word $weight$ in the UMLS has two possible CUIs, "C0005912: Body Weight" and "C0699807: Weight", each having the ST "Quantitative Concept". Humphrey et al.'s (2006) approach relies on the concepts having different STs therefore is unable to disambiguate between these two concepts.

Currently, most word sense disambiguation approaches focus on lexical sample disambiguation which only attempts to disambiguate a predefined set of words. This type of disambiguation is not practical for large scale systems. All-words disambiguation approaches disambiguate all ambiguous words in a running text making them practical for large scale systems. Unlike SenseClusters, Humphrey, et al. (2006) and our approach can be

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 49–54,
Columbus, June 2008.

used to perform all-words disambiguation.

In the following sections, we first discuss related work. We then discuss our approach, experiments and results. Lastly, we discuss our conclusions and future work.

2 Related Work

There has been previous work on word sense disambiguation in the biomedical domain. Leroy and Rindflesch (2005) introduce a supervised approach that uses the UMLS STs and their semantic relations of the words surrounding the target word as features into a Naive Bayes classifier. Joshi et al. (2005) introduce a supervised approach that uses unigrams and bigrams surrounding the target word as features into a Support Vector Machine. A unigram is a single content word that occurs in a window of context around the target word. A bigram is an ordered pair of content words that occur in a window of context around the target word. McInnes et al. (2007) introduce a supervised approach that uses CUIs of the words surrounding the target word as features into a Naive Bayes classifier.

Humphrey et al. (2006) introduce an unsupervised vector approach using Journal Descriptor (JD) Indexing (JDI) which is a ranking algorithm that assigns JDs to journal titles in MEDLINE. The authors apply the JDI algorithm to STs with the assumption that each possible concept has a distinct ST. In this approach, an ST vector is created for each ST by extracting associated words from the UMLS. A target word vector is created using the words surrounding the target word. The JDI algorithm is used to obtain a score for each word-JD and ST-JD pair using the target word and ST vectors. These pairs are used to create a word-ST table using the cosine coefficient between the scores. The cosine scores for the STs of each word surrounding the target word are averaged and the concept associated with the ST that has the highest average is assigned to the target word.

3 Vector Approaches

Patwardhan and Pedersen (2006) introduce a vector measure to determine the relatedness between pairs of concepts. In this measure, a co-occurrence matrix of all words in a given corpus is created containing how often they occur in the same window of context with each other. A gloss vector is then created for each concept containing the word vector for each word in the concepts definition (or gloss). The cosine between the two gloss vectors is computed to determine the concepts relatedness.

SenseClusters [1] is an unsupervised knowledge-lean word sense disambiguation package The package uses clustering algorithms to group similar instances of target words and label them with the appropriate sense. The clustering algorithms include Agglomerative, Graph partitional-based, Partitional biased agglomerative and Direct k-way clustering. The clustering can be done in either vector space where the vectors are clustered directly or similarity space where vectors are clustered by finding the pair-wise similarities among the contexts. The feature options available are first and second-order co-occurrence, unigram and bigram vectors. First-order vectors are highly frequent words, unigrams or bigrams that co-occur in the same window of context as the target word. Second-order vectors are highly frequent words that occur with the words in their respective first order vector.

We compare our approach to SenseClusters v0.95 using direct k-way clustering with the I2 clustering criterion function and cluster in vector space. We experiment with first-order unigrams and second-order bigrams with a Log Likelihood Ratio greater than 3.84 and the exact and gap cluster stopping parameters (Purandare and Pedersen, 2004; Kulkarni and Pedersen, 2005).

4 Our Approach

Our approach has three stages: i) we create a the feature vector for the target word ($instance\ vector$) and each of its possible concepts ($concept\ vectors$) using SenseClusters, ii) we calculate the cosine between the instance vector and each of the concept vectors, and iii) we assign the concept whose concept vector is the closest to the instance vector to the target word.

To create the the instance vector, we use the words that occur in the same abstract as the target word as features. To create the concept vector, we explore four different context descriptions of a possible concept to use as features. Since each possible concept

[1] http://senseclusters.sourceforge.net/

has a corresponding CUI in the UMLS, we explore using: i) the words in the concept's CUI definition, ii) the words in the definition of the concept's ST definition, iii) the words in both the CUI and ST definitions, and iv) the words in the CUI definition unless one does not exist then the words in its ST definition.

We explore using the same feature vector parameters as in the SenseCluster experiments: i) first-order unigrams, and ii) second-order bigram. We also explore using a more judicious approach to determine which words to include in the feature vectors. One of the problems with an unsupervised vector approach is its susceptibility to noise. A word frequently seen in a majority of instances may not be useful in distinguishing between different concepts. To alleviate this problem, we create an individualized stoplist for each target word using the inverse document frequency (IDF). We calculate the IDF score for each word surrounding the target word by taking the log of the number of documents in the training data divided by the number of documents the term has occurred in the dataset. We then extract those words that obtain an IDF score under the threshold of one and add them to our basic stoplist to be used when determining the appropriate sense for that specific target word.

5 Data

5.1 Training Data

We use the abstracts from the 2005 Medline Baseline as training data. The data contains 14,792,864 citations from the 2005 Medline repository. The baseline contains 2,043,918 unique tokens and 295,585 unique concepts.

5.2 NLM-WSD Test Dataset

We use the National Library of Medicine's Word Sense Disambiguation (NLM-WSD) dataset developed by (Weeber et al., 2001) as our test set. This dataset contains 100 instances of 50 ambiguous words from 1998 MEDLINE abstracts. Each instance of a target word was manually disambiguated by 11 human evaluators who assigned the word a CUI or "None" if none of the CUIs described the concept. (Humphrey et al., 2006) evaluate their approach using a subset of 13 out of the 50 words whose majority sense is less than 65% and whose possible concepts do not have the same ST. Instances tagged as "None" were removed from the dataset. We evaluate our approach using these same words and instances.

5.3 Conflate Test Dataset

To test our algorithm on a larger biomedical dataset, we are creating our own dataset by conflating two or more unambiguous words from the 2005 Medline Baseline. We determine which words to conflate based on the following criteria: i) the words have a single concept in the UMLS, ii) the words occur approximately the same number of times in the corpus, and iii) the words do not co-occur together.

We create our dataset using $name\text{-}conflate$ [2] to extract instances containing the conflate words from the 2005 Medline Baseline. Table 4 shows our current set of conflated words with their corresponding number of test (test) and training (train) instances. We refer to the conflated words as their pseudowords throughout the paper.

6 Experimental Results

In this section, we report the results of our experiments. First, we compare the results of using the IDF stoplist over a basic stoplist. Second, we compare the results of using the different context descriptions. Third, we compare our approach to SenseClusters and Humphrey et al. (2006) using the NLM-WSD dataset. Lastly, we compare our approach to SenseClusters using the conflated dataset.

In the following tables, CUI refers to the CUI definition of the possible concept as context, ST refers to using the ST definition of the possible concept as context, CUI+ST refers to using both definitions as context, and CUI→ST refers to using the CUI definition unless if one doesn't exist then using ST definition. Maj. refers to the "majority sense" baseline which is accuracy that would be achieved by assigning every instance of the target word with the most frequent sense as assigned by the human evaluators.

6.1 Stoplist Results

Table 2 shows the overall accuracy of our approach using the basic stoplist and the IDF stoplist on the

[2] http://www.d.umn.edu/ tpederse/namedata.html

target word	Unigram				Bigram			
	CUI	ST	CUI+ST	CUI→ST	CUI	ST	CUI+ST	CUI→ST
adjustment	44.57	31.61	46.74	44.57	**47.83**	38.04	27.17	**47.83**
blood pressure	39.39	34.34	41.41	38.38	43.43	27.27	**47.47**	38.38
degree	3.13	**70.31**	**70.31**	**70.31**	3.13	48.44	48.44	48.44
evaluation	50.51	50.51	53.54	51.52	50.51	**54.55**	52.53	51.52
growth	**63.64**	51.52	42.42	**63.64**	**63.64**	51.52	48.48	**63.64**
immunosuppression	50.51	46.46	50.51	50.51	43.43	**57.58**	48.48	43.43
mosaic	0	33.33	27.08	**37.50**	0	28.13	22.92	22.92
nutrition	28.41	34.09	35.23	25.00	38.64	**39.77**	36.36	37.50
radiation	57.73	44.78	58.76	57.73	**60.82**	28.36	**60.82**	**60.82**
repair	74.63	25.00	41.79	37.31	**76.12**	54.69	44.78	41.79
scale	32.81	48.00	42.19	51.56	0	18.00	95.31	**96.88**
sensitivity	6.00	**50.56**	48.00	48.00	8.00	44.94	18.00	18.00
white	48.31	38.61	46.07	**49.44**	44.94	38.16	43.82	**49.44**
average	38.43	43.01	46.46	**48.11**	36.96	40.73	45.74	47.74

Table 1: Accuracy of Our Approach using Different Context Descriptions

NLM-WSD dataset using each of the different context descriptions described above. The results show an approximately a 2% higher accuracy over using the basic stoplist. The exception is when using the CUI context description; the accuracy decreased by approximately 2% when using the unigram feature set and approximately 1% when using the bigram feature set.

context	Basic stoplist		IDF stoplist	
	unigram	bigram	unigram	bigram
CUI	**41.02**	**37.68**	38.43	36.96
ST	42.74	37.14	**43.01**	**40.73**
CUI+ST	44.13	42.71	**46.46**	**45.74**
CUI→ST	46.61	45.58	**48.11**	**47.74**

Table 2: Accuracy of IDF stoplist on the NLM-WSD dataset

6.1.1 Context Results

Table 1 shows the results of our approach using the CUI and ST definitions as context for the possible concepts on the NLM-WSD dataset and Table 4 shows similar results using the conflate dataset.

On the NLM-WSD dataset, the results show a large difference in accuracy between the contexts on a word by word basis making it difficult to determine which of the context description performs the best. The unigram results show that CUI→ST and CUI+ST obtain the highest accuracy for five words, and CUI and ST obtain the highest accuracy for one word. The bigram results show that CUI→ST and CUI obtains the highest accuracy for two words, ST obtains the highest accuracy for four words, and CUI+ST obtains the highest accuracy for one word. The overall results show that using unigrams with the context description CUI→ST obtains the highest overall accuracy.

On the conflated dataset, the pseudowords a_a, a_o, d_d and e_e have a corresponding CUI definition for each of their possible concepts therefore the accuracy for CUI and CUI→ would be the same for these datasets and is not reported. The pseudowords a_a_i, x_p_p and d_a_m_e do not have a CUI definitions for each of their possible concepts. The results show that CUI obtained the highest accuracy for six out of the seven datasets and CUI→ST obtained the highest accuracy for one. These experiments were run using the unigram feature.

6.2 NLM-WSD Results

Table 3 shows the accuracy of the results obtained by our unsupervised vector approach using the CUI→ST context description, SenseClusters, and the results reported by Humphrey et al. (2006).

As seen with the context description results, there exists a large difference in accuracy on a word by word basis between the approaches. The results show that Humphrey et al. (2006) report a higher overall accuracy compared to SenseClusters and our approach. Although, Humphrey et al. (2006) performed better for 5 out of the 13 words where as SenseClusters performed better for 9. The unigram feature set with gap cluster stopping returned the highest overall accuracy for SenseClusters. The number of clusters for all of the gap cluster stopping experiments were two except for $growth$ which returned one. For our approach, the unigram feature set returned the highest overall accuracy.

target word	senses	Maj.	Humphrey	SenseClusters				Our Approach	
			et al. 2006	exact cluster stopping		gap cluster stopping		CUI→ST	
				unigram	bigram	unigram	bigram	unigram	bigram
adjustment	3	66.67	**76.67**	49.46	38.71	55.91	45.16	44.57	47.83
blood pressure	3	**54.00**	41.79	40.00	46.00	51.00	**54.00**	38.38	38.38
degree	2	96.92	**97.73**	53.85	55.38	53.85	55.38	70.31	48.44
evaluation	2	50.00	59.70	**66.00**	50.00	**66.00**	50.00	51.52	51.52
growth	2	63.00	**70.15**	66.00	52.00	66.00	63.00	63.64	63.64
immunosuppression	2	59.00	74.63	67.00	**80.00**	67.00	**80.00**	50.51	43.43
mosaic	2	53.61	67.69	**72.22**	58.57	61.86	50.52	37.50	22.92
nutrition	2	**50.56**	35.48	40.45	47.19	44.94	41.57	25.00	37.50
radiation	2	62.24	**78.79**	69.39	56.12	69.39	56.12	57.73	60.82
repair	2	76.47	86.36	**86.76**	73.53	86.76	73.53	37.31	41.79
scale	2	**100.0**	60.47	**100.0**	**100.0**	**100.0**	**100.0**	51.56	96.88
sensitivity	2	**96.08**	82.86	41.18	41.18	52.94	54.90	48.00	18.00
white	2	54.44	55.00	**80.00**	53.33	**80.00**	53.33	49.44	49.44
average		67.92	**68.26**	64.02	57.85	65.82	59.81	48.11	47.74

Table 3: Accuracy of Approaches using the NLM-WSD Dataset

target word	pseudo-word	test	train	Maj.	Sense Clusters	Our Approach			
						CUI	ST	CUI+ST	CUI→ST
actin-antigens	a_a	33193	298723	63.44	**91.30**	53.95	44.81	54.17	
angiotensin II-olgomycin	a_o	5256	47294	93.97	**56.76**	16.62	20.68	17.73	
dehydrogenase-diastolic	d_d	22606	203441	58.57	**95.85**	45.78	43.94	45.70	
endogenous-extracellular matrix	e_e	19820	178364	79.92	71.21	**74.34**	65.37	73.37	
allogenic-arginine-ischemic	a_a_i	22915	206224	57.16	**69.03**	47.68	24.60	33.77	32.07
X chromosome-peptide-plasmid	x_p_p	46102	414904	74.61	**66.21**	20.04	31.60	42.89	42.98
diacetate-apamin-meatus-enterocyte	d_a_m_e	1358	12212	25.95	**74.23**	28.87	24.08	26.07	22.68

Table 4: Accuracy of Approaches using the Conflate Dataset

6.3 Conflate Results

Table 4 shows the accuracy of the results obtained by our approach and SenseClusters. The results show that SenseClusters returns a higher accuracy than our approach except for the e_e dataset.

7 Discussion

We report the results for four experiments in this paper: i) the results of using the IDF stoplist over a basic stoplist, ii) the results of our approach using different context descriptions of the possible concepts of a target word, iii) the results of our approach compared to SenseClusters and Humphrey et al. (2006) using the NLM-WSD dataset, and iv) the results of our approach compared to SenseClusters using the conflated dataset.

The results of using an individualized IDF stoplist for each target word show an improvement over using the basic stoplist. The results of our approach using different context descriptions show that for the NLM-WSD dataset the large differences in accuracy makes it unclear which of the context descriptions performed the best. On the conflated dataset, adding the ST definition to the context description improved the accuracy of only one pseudoword. When comparing our approach to Humphrey et al. (2006) and SenseClusters, our approach did not return a higher accuracy.

When analyzing the data, we found that there does not exist a CUI definition for a large number of possible concepts. Table 5 shows the number of words in the CUI and ST definitions for each concept in the NLM-WSD dataset. Only four target words have a CUI definition for each possible concept. We also found the concept definitions vary widely in length. The CUI definitions in the UMLS come from a variety of sources and there may exist more than one definition per source. Unlike CUI definitions, there does exist an ST definition for each possible concept. The ST definitions come from the same source and are approximately the same length but they are a broad categorization. We believe this makes them too coarse grained to provide descriptive enough information about their associated concepts.

This can also be seen when analyzing the conflate datasets. The conflate dataset d_a_m_e is missing two definition which is a contributing factor to its low accuracy for CUI. Adding the ST definition

target word	CUI Definition			ST Definition		
	c1	c2	c3	c1	c2	c3
adjustment	41	9	48	31	19	10
blood pressure	26	18	0	20	31	22
degree	0	0		15	23	
evaluation	54	0		33	17	
growth	91	91		20	19	
immunosuppression	130	41		30	20	
mosaic	0	38	0	10	10	23
nutrition	152	152	0	10	31	30
radiation	71	207		14	30	
repair	0	51		30	20	
scale	0	10	144	47	23	8
sensitivity	0	0	0	25	50	22
white	0	60		15	28	

Table 5: Number of words in CUI and ST Definitions of Possible the Concepts in the NLM-WSD Dataset

though did not provide enough distinctive information to distinguish between the possible concepts.

8 Conclusions and Future Work

This paper introduces an unsupervised vector approach to disambiguate words in biomedical text using contextual information from the UMLS. Our approach makes disambiguation decisions for words that have the same ST unlike Humphrey et al. (2006). We believe that our approach shows promise and leads us to our goal of exploring the use of biomedical knowledge sources.

In the future, we would also like to increase the size of our conflated dataset and possibly create a biomedical all-words disambiguation test set to test our approach. Unlike SenseClusters, our approach can be used to perform all-words disambiguation. For example, given the sentence: *His weight has fluctuated during the past month.* We first create a instance vector containing $fluctuated$, $past$ and $months$ for the word $weight$ and a concept vector for each of its possible concepts, "C0005912: Body Weight" and "C0699807: Quantitative Concept" using their context descriptions. We then calculate the cosine between the instance vector and each of the two concept vectors. The concept whose vector has the smallest cosine score is assigned to $weight$. We then repeat this process for $fluctuated$, $past$ and $months$.

We also plan to explore using different contextual information to improve the accuracy of our approach. We are currently exploring using co-occurrence and relational information about the possible CUIs in the UMLS. Our IDF stoplist experiments show promise, we are planning to explore other measures to determine which words to include in the stoplist as well as a way to automatically determine the threshold.

Acknowledgments

The author thanks Ted Pedersen, John Carlis and Siddharth Patwardhan for their comments.

Our experiments were conducted using CuiTools v0.15, which is freely available from http://cuitools.sourceforge.net.

References

S.M. Humphrey, W.J. Rogers, H. Kilicoglu, D. Demner-Fushman, and T.C. Rindflesch. 2006. Word sense disambiguation by selecting the best semantic type based on journal descriptor indexing: Preliminary experiment. *Journal of the American Society for Information Science and Technolology*, 57(1):96–113.

M. Joshi, T. Pedersen, and R. Maclin. 2005. A comparative study of support vectors machines applied to the supervised word sense disambiguation problem in the medical domain. In *Proceedings of 2nd Indian International Conference on AI*, pages 3449–3468, Dec.

A. Kulkarni and T. Pedersen. 2005. SenseClusters: unsupervised clustering and labeling of similar contexts. In *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*, pages 105–108, June.

G. Leroy and T.C. Rindflesch. 2005. Effects of information and machine learning algorithms on word sense disambiguation with small datasets. *International Journal of Medical Info.*, 74(7-8):573–85.

B. McInnes, T. Pedersen, and J. Carlis. 2007. Using umls concept unique identifiers (cuis) for word sense disambiguation in the biomedical domain. In *Proceedings of the Annual Symposium of the American Medical Informatics Association*, pages 533–37, Chicago, IL, Nov.

S. Patwardhan and T. Pedersen. 2006. Using WordNet-based Context Vectors to Estimate the Semantic Relatedness of Concepts. In *Proceedings of the EACL 2006 Workshop Making Sense of Sense - Bringing Computational Linguistics and Psycholinguistics Together*, volume 1501, pages 1–8, Trento, Italy, April.

A. Purandare and T. Pedersen. 2004. Word sense discrimination by clustering contexts in vector and similarity spaces. In *Proceedings of the Conference on CoNLL*, pages 41–48.

M. Weeber, J.G. Mork, and A.R. Aronson. 2001. Developing a test collection for biomedical word sense disambiguation. In *Proceedings of the American Medical Informatics Association Symposium*, pages 746–750.

A Subcategorization Acquisition System for French Verbs

Cédric Messiant
Laboratoire d'Informatique de Paris-Nord
CNRS UMR 7030 and Université Paris 13
99, avenue Jean-Baptiste Clément, F-93430 Villetaneuse France
cedric.messiant@lipn.univ-paris13.fr

Abstract

This paper presents a system capable of automatically acquiring subcategorization frames (SCFs) for French verbs from the analysis of large corpora. We applied the system to a large newspaper corpus (consisting of 10 years of the French newspaper 'Le Monde') and acquired subcategorization information for 3267 verbs. The system learned 286 SCF types for these verbs. From the analysis of 25 representative verbs, we obtained 0.82 precision, 0.59 recall and 0.69 F-measure. These results are comparable with those reported in recent related work.

1 Introduction

Many Natural Language Processing (NLP) tasks require comprehensive lexical resources. Handcrafting large lexicons is labour-intensive and error-prone. A growing body of research focuses therefore on automatic acquisition of lexical resources from text corpora.

One useful type of lexical information for NLP is the number and type of the arguments of predicates. These are typically expressed in simple syntactic frames called subcategorization frames (SCFs). SCFs can be useful for many NLP applications, such as parsing (John Carroll and Briscoe, 1998) or information extraction (Surdeanu et al., 2003). Automatic acquisition of SCFs has therfore been an active research area since the mid-90s (Manning, 1993; Brent, 1993; Briscoe and Carroll, 1997).

Comprehensive subcategorization information is currently not available for most languages. French is one of these languages: although manually built syntax dictionaries do exist (Gross, 1975; van den Eynde and Mertens, 2006; Sagot et al., 2006) none of them are ideal for computational use and none also provide frequency information important for statistical NLP.

We developed *ASSCI*, a system capable of extracting large subcategorization lexicons for French verbs from raw corpus data. Our system is based on a approach similar to that of the well-known Cambridge subcategorization acquisition system for English (Briscoe and Carroll, 1997; Preiss et al., 2007). The main difference is that unlike the Cambridge system, our system does not employ a set of predefined SCF types, but learns the latter dynamically from corpus data.

We have recently used *ASSCI* to acquire *LexSchem* – a large subcategorization lexicon for French verbs – from a raw journalistic corpus. and have made the resulting resource freely available to the community on the web (Messiant et al., 2008).

We describe our SCF acquisition system in section 2 and explain the acquisition of a large subcategorization lexicon for French and its evaluation in section 3. We finally compare our study with work previously achieved for English and French in section 4.

2 ASSCI: The Acquisition System

Our SCF acquisition system takes as input corpus data and produces a list of frames for each verb that occurred more than 200 times in the corpus. It the first system that automatically induces a large-scale SCF information from raw corpus data for French.

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 55–60,
Columbus, June 2008.

Previous experiments focussed on a limited set of verbs (Chesley and Salmon-Alt, 2006), or were based on treebanks or on substantial manual work (Gross, 1975; Kupść, 2007).

The system works in three steps:

1. verbs and surrounding phrases are extracted from parsed corpus data;
2. tentative SCFs are built dynamically, based on morpho-syntactic information and relations between the verb and its arguments;
3. a statistical filter is used to filter out incorrect frames.

2.1 Preprocessing

When aiming to build a large lexicon for general language, the input data should be large, balanced and representative enough. Our system tags and lemmatizes input data using *TreeTagger* (Schmid, 1994) and then syntactically analyses it using *Syntex* (Bourigault et al., 2005). The *TreeTagger* is a statistical, language independent tool for the automatic annotation of part-of-speech and lemma information. *Syntex* is a shallow parser for extracting lexical dependencies (such as adjective/noun or verb/noun dependencies). *Syntex* obtained the best precision and F-measure for written French text in the recent EASY evaluation campaign[1].

The dependencies extracted by the parser include both arguments and adjuncts (such as location or time phrases). The parsing strategy is based on heuristics and statistics only. This is ideal for us since no lexical information should be used when the task is to acquire it. *Syntex* works on the general assumption that the word on the left side of the verb is the subject, where as the word on the right is the object. Exceptions to this assumption are dealt with a set of rules.

```
(2) Ces propriétaires exploitants
achètent ferme le carburant  la
compagnie .
(These owners buy fast the fuel to
the company.)
```

(3) is the preprocessed *ASSCI* input for sentence (2) (after the *TreeTagger* annotation and *Syntex*'s analysis).

```
(3) DetMP|ce|Ces|1|DET;3|
AdjMP|propriétaire|propriétaires|2|ADJ;3|
NomMP|exploitant|exploitants|3||DET;1,ADJ;2
VCONJP|acheter|achètent|4||ADV;5,OBJ;7,PREP;8
Adv|ferme|ferme|5|ADV;11|
DetMS|le|le|6|DET;7|
NomMS|carburant|carburant|7|OBJ;4|DET;6
Prep|à|à|8|PREP;4|NOMPREP;10
DetFS|le|la|9|DET;10|
NomFS|compagnie|compagnie|10|NOMPREP;8|DET;9
Typo|.|.|11||
```

2.2 Pattern Extractor

The pattern extraction module takes as input the syntactic analysis of *Syntex* and extracts each verb which is sufficiently frequent (the minimum of 200 corpus occurrences) in the syntactically analysed corpus data, along with surrounding phrases. In some cases, this module makes deeper use of the dependency relations in the analysis. For example, when a preposition is part of the dependencies, the pattern extractor examines whether this preposition is followed by a noun phrase or an infinitive clause.

(4) is the output of the pattern extractor for (3).

```
(4) VCONJP|acheter
NomMS|carburant|OBJ_Prep|à+SN|PREP
```

Note that +SN marks that the *"à"* preposition is followed by a noun phrase.

2.3 SCF Builder

The next module examines the dependencies according to their syntactic category (e.g., noun phrase) and their relation to the verb (e.g., object), if any. It constructs frames dynamically from the following features: a nominal phrase; infinitive clause; prepositional phrase followed by a noun phrase; prepositional phrase followed by an infinitive clause; subordinate clause and adjectival phrase. If the verb has no dependencies, its SCF is "intransitive" (INTRANS). The number of occurrences for each

[1] http://www.limsi.fr/Recherche/CORVAL/easy
The scores and ranks of *Syntex* at this evaluation campaign are available at http://w3.univ-tlse2.fr/erss/textes/pagespersos/bourigault/syntex.html#easy

SCF and the total number of occurrences with each verb are recorded.

This dynamic approach to SCF learning was adopted because no sufficiently comprehensive list of SCFs was available for French (most previous work on English (e.g., (Preiss et al., 2007)) employs a set of predefined SCFs because a relatively comprehensive lists are available for English).

The SCF candidate built for sentence (2) is shown in (5)[2].

(5) SN_SP[à+SN]

2.4 SCF Filter

The final module filters the SCF candidates. A filter is necessary since the output of the second module is noisy, mainly because of tagging and parsing errors but also because of the inherent difficulty of argument-adjunct distinction which ideally requires access to the lexical information we aim to acquire, along with other information and criteria which current NLP systems (and even humans) find it difficult to identify. Several previous works (e.g., (Briscoe and Carroll, 1997; Chesley and Salmon-Alt, 2006)) have used binomial hypothesis testing for filtering. Korhonen et al. (2000) proposes to use the maximum likelihood estimate and shows that this method gives better results than the filter based on binomial hypothesis testing. This method employs on a simple threshold over the relative frequencies of SCFs candidates. (The maximum likehood estimate is still an option in the current Cambridge system but an improved version calculates it specific to different SCFs - a method which we left for future work).

The relative frequency of the SCF *i* with the verb *j* is calculated as follows:

$$rel\_freq(scf_i, verb_j) = \frac{|scf_i, verb_j|}{|verb_j|}$$

$|scf_i, verb_j|$ is the number of occurrences of the SCF *i* with the verb *j* and $|verb_j|$ is the total number of occurrences of the verb *j* in the corpus.

These estimates are compared with the threshold value to filter out low probability frames for each verb. The effect of the choice of the threshold on the results is discussed in section 3.

[2]SN stands for a noun phrase and SP for a prepositional phrase

3 Experimental Evaluation

3.1 Corpus

In order to evaluate our system on a large corpus, we gathered ten years of the French newspaper *Le Monde* (two hundred millions words). It is one of the largest corpus for French and "clean" enough to be easily and efficiently parsed. Because our aim was to acquire a large general lexicon, we require the minimum of 200 occurrences per each verb we analysed using this system.

3.2 LexSchem: The Acquired Lexicon

3267 verbs were found with more than 200 occurrences in the corpus. From the data for these verbs, we induced 286 distinct SCF types. We have made the extracted lexicon freely available on the web (http://www-lipn.univ-paris13.fr/~messiant/lexschem.html) under the LGPL-LR (Lesser General Public License For Linguistic Resources) license. An interface which enables viewing the SCFs acquired for each verb and the verbs taking different SCFs is also available at the same address. For more details of the lexicon and its format, see (Messiant et al., 2008).

3.3 Gold Standard

Direct evaluation of subcategorization acquisition performance against a *gold standard* based on a manmade dictionary is not ideal (see e.g. (Poibeau and Messiant, 2008)). However, this method is still the easiest and fastest way to get an idea of the performance of the system. We built a *gold standard* using the SCFs found in the *Trésor de la Langue Française Informatisé (TFLI)*, a large French dictionary available on the web[3]. We evaluated 25 verbs listed in Appendix to evaluate our system. These verbs were chosen for their heterogeneity in terms of semantic and syntactic features, but also because of their varied frequency in the corpus (from 200 to 100.000 occurences).

3.4 Evaluation Measures

We calculated type precision, type recall and F-measure for these 25 verbs. We obtain the best results (0.822 precision, 0.587 recall and 0.685 f-measure) with the MLE threshold of 0.032 (see fig-

[3]http://atilf.atilf.fr/

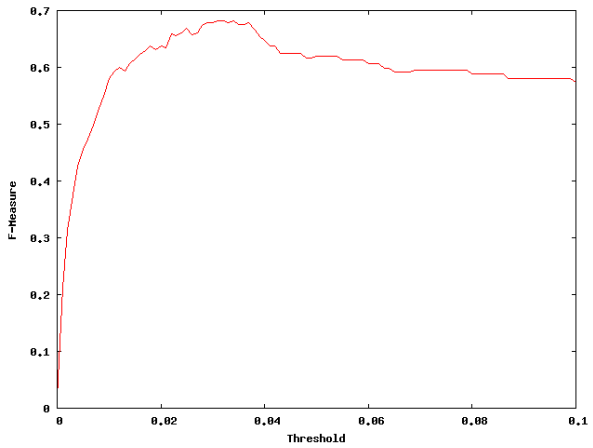


Figure 1: The relation of the threshold on the F-Measure

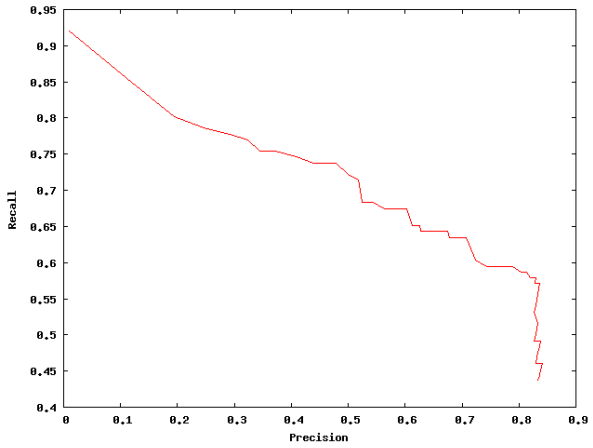


Figure 2: The relation between precision and recall

ure 1). Figure 2 shows that even by substantially lowering recall we cannot raise precision over 0.85.

Table 1 shows a comparison of three versions of *ASSCI* for our 25 verbs:

- Unfiltered: the unfiltered output of *ASSCI*;
- ASSCI-1: one single threshold fixed to 0.0325;
- ASSCI-2: one INTRANS-specific threshold (0.08) and the 0.0325-threshold for all other cases.

These results reveal that the unfiltered version of the lexicon is very noisy indeed (0.01 precision).

System	Precision	Recall	F-Measure
Unfiltered	0.010	0.921	0.020
ASSCI-1	0.789	0.595	0.679
ASSCI-2	0.822	0.587	0.685

Table 1: Comparison of different versions of *ASSCI*

A simple threshold on the relative frequencies improves the results dramatically (*ASSCI-1*).

Each step of the acquisition process generates errors. For example, some nouns are tagged as a `verb` by *TreeTagger* (e.g., in the phrase *"Le programme d'armement (weapons program)"*, "*programme*" is tagged `verb`). *Syntex* generates errors when identifying dependencies: in some cases, it fails to identify relevant dependencies; in other cases incorrect dependencies are generated. The SCF builder is another source of error because of the ambiguity or the lack of sufficient information to build some frames (e.g. those involving pronouns). Finally, the filtering module rejects some correct SCFs and accept some incorrect ones. We could reduce these errors by improving the filtering method or refining the thresholds.

Many of the errors involve intransitive SCFs. We tried to address this problem with an INTRANS-specific threshold which is higher than others (see the results for *ASSCI-2*). This improves the precision of the system slightly but does not substantially reduce the number of false negatives. The intransitive form of verbs is very frequent in corpus data but it doesn't appear in the *gold standard*. A better evaluation (e.g., a gold standard based on manual analysis of the corpus data and annotation for SCFs) should not yield these errors. In other cases (e.g. interpolated clauses), the parser is incapable of finding the dependencies. In subsequent work we plan to use an improved version of *Syntex* which deals with this problem.

Our results (*ASSCI-2*) are similar with those obtained by the only directly comparable work for French (Chesley and Salmon-Alt, 2006) (0.87 precision and 0.54 recall). However, the lexicons show still room for improvement, especially with recall. In addition to the improvements in the method and evaluation suggested above, we plan to evaluate whether lexicons resulting from our system are use-

ful for NLP tasks and applications. For example, John Carroll *& al.* shows that a parser can be significantly improved by using a SCF lexicon despite a high error rate (John Carroll and Briscoe, 1998).

4 Related Work

4.1 Manual or Semi-Automatic Work

Most previous subcategorization lexicons for French were built manually. For example, Maurice Gross built a large French dictionnary called "*Les Tables du LADL*" (Gross, 1975). This dictionary is not easy to employ for NLP use but work in progress is aimed at addressing this problem (Gardent et al., 2005). The *Lefff* is a morphological and syntactic lexicon that contains partial subcategorization information (Sagot et al., 2006), while *Dicovalence* is a manually built valency dictionnary based on the pronominal approach (van den Eynde and Blanche-Benveniste, 1978; van den Eynde and Mertens, 2006). There are also lexicons built using semi-automatic approaches e.g., the acquisition of subcategorization information from treebanks (Kupść, 2007).

4.2 Automatic Work

Experiments have been made on the automatic acquisition of subcategorization frames since mid 1990s (Brent, 1993; Briscoe and Carroll, 1997). The first experiments were performed on English but since the beginning of 2000s the approach has been successfully applied to various other languages. For example, (Schulte im Walde, 2002) has induced a subcategorization lexicon for German verbs from a lexicalized PCFG. Our approach is quite similar to the work done in Cambridge. The Cambridge system has been regularly improved and evaluated; and it represents the state-of-the-art perfomance on the task (Briscoe and Carroll, 1997; Korhonen et al., 2000; Preiss et al., 2007). In the latest paper, the authors show that the method can be successfully applied to acquire SCFs not only for verbs but also for nouns and adjectives (Preiss et al., 2007). A major difference between these related works and ours is the fact that we do not use a predefined set of SCFs. Of course, the number of frames depends on the language, the corpus, the domain and the information taken into account (for example, (Preiss et al., 2007) used a list of 168 predefined frames for English which abstract over lexically-governed prepositions).

As far as we know, the only directly comparable work on subcategorization acquisition for French is (Chesley and Salmon-Alt, 2006) who propose a method for acquiring SCFs from a multi-genre corpus in French. Their work relies on the VISL parser which have an "unevaluated (and potentially high) error rate" while our system relies on *Syntex* which is, according to the *EASY evaluation campaign*, the best parser for French (as evaluated on general newspaper corpora). Additionally, we acquired a large subcategorization lexicon (available on the web) (286 distinct SCFs for 3267 verbs) whereas (Chesley and Salmon-Alt, 2006) produced only 27 SCFs for 104 verbs and didn't produce any lexicon for public release.

5 Conclusion

We have introduced a system which we have developed for acquiring large subcategorization lexicons for French verbs. When the system was applied to a large French newspaper corpus, it produced a lexicon of 286 SCFs corresponding to 3267 verbs. We evaluated this lexicon by comparing the SCFs it produced for 25 test verbs to those included in a manually built dictionary and obtained promising results. We made the automatically acquired lexicon freely available on the web under the LGPL-LR license (and through a web interface).

Future work will include improvements of the filtering module (using e.g. SCF-specific thresholds or statistical hypothesis testing) and exploration of task-based evaluation in the context of practical NLP applications and tasks such as the acquisition of semantic classes from the SCFs (Levin, 1993).

Acknowledgements

Cédric Messiant's PhD is funded by a DGA/CNRS Grant. The research presented in this paper was also supported by the ANR MDCO 'CroTal' project and the British Council and the French Ministry of Foreign Affairs -funded 'Alliance' grant.

References

Didier Bourigault, Marie-Paule Jacques, Cécile Fabre, Cécile Frérot, and Sylwia Ozdowska. 2005. Syntex,

analyseur syntaxique de corpus. In *Actes des 12èmes journées sur le Traitement Automatique des Langues Naturelles*, Dourdan.

Michael R. Brent. 1993. From Grammar to Lexicon: Unsupervised Learning of Lexical Syntax. *Computational Linguistics*, 19:203–222.

Ted Briscoe and John Carroll. 1997. Automatic Extraction of Subcategorization from Corpora. In *Proceedings of the 5th ACL Conference on Applied Natural Language Processing*, pages 356–363, Washington, DC.

Paula Chesley and Susanne Salmon-Alt. 2006. Automatic extraction of subcategorization frames for French. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Genua (Italy).

Claire Gardent, Bruno Guillaume, Guy Perrier, and Ingrid Falk. 2005. Maurice Gross' Grammar Lexicon and Natural Language Processing. In *2nd Language and Technology Conference*, Poznan.

Maurice Gross. 1975. *Méthodes en syntaxe*. Hermann, Paris.

Guido Minnen John Carroll and Ted Briscoe. 1998. Can subcategorisation probabilities help a statistical parser? In *Proceedings of the 6th ACL/SIGDAT Workshop on Very Large Corpora*, Montreal (Canada).

Anna Korhonen, Genevieve Gorrell, and Diana McCarthy. 2000. Statistical filtering and subcategorization frame acquisition. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, Hong Kong.

Anna Kupść. 2007. Extraction automatique de cadres de sous-catégorisation verbale pour le français à partir d'un corpus arboré. In *Actes des 14èmes journées sur le Traitement Automatique des Langues Naturelles*, Toulouse, June.

Beth Levin. 1993. *English Verb Classes and Alternations: a preliminary investigation*. University of Chicago Press, Chicago and London.

Christopher D. Manning. 1993. Automatic Acquisition of a Large Subcategorization Dictionary from Corpora. In *Proceedings of the Meeting of the Association for Computational Linguistics*, pages 235–242.

Cédric Messiant, Anna Korhonen, and Thierry Poibeau. 2008. LexSchem : A Large Subcategorization Lexicon for French Verbs. In *Language Resources and Evaluation Conference (LREC)*, Marrakech.

Thierry Poibeau and Cédric Messiant. 2008. Do We Still Need Gold Standard For Evaluation ? In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Marrakech.

Judita Preiss, Ted Briscoe, and Anna Korhonen. 2007. A System for Large-Scale Acquisition of Verbal, Nominal and Adjectival Subcategorization Frames from Corpora. In *Proceedings of the Meeting of the Association for Computational Linguistics*, pages 912–918, Prague.

Benoît Sagot, Lionel Clément, Eric de La Clergerie, and Pierre Boullier. 2006. The Lefff 2 syntactic lexicon for French: architecture, acquisition, use. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Genua (Italy).

Helmut Schmid. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *International Conference on New Methods in Language Processing*, Manchester, UK. unknown.

Sabine Schulte im Walde. 2002. A Subcategorisation Lexicon for German Verbs induced from a Lexicalised PCFG. In *Proceedings of the 3rd Conference on Language Resources and Evaluation*, volume IV, pages 1351–1357, Las Palmas de Gran Canaria, Spain.

Mihai Surdeanu, Sanda M. Harabagiu, John Williams, and Paul Aarseth. 2003. Using Predicate-Argument Structures for Information Extraction. In *Proceedings of the Association of Computational Linguistics (ACL)*, pages 8–15.

Karel van den Eynde and Claire Blanche-Benveniste. 1978. Syntaxe et mécanismes descriptifs : présentation de l'approche pronominale. *Cahiers de Lexicologie*, 32:3–27.

Karel van den Eynde and Piet Mertens. 2006. *Le dictionnaire de valence Dicovalence : manuel d'utilisation*. Manuscript, Leuven.

Appendix — List of test verbs

compter	donner	apprendre
chercher	possder	comprendre
concevoir	proposer	montrer
rendre	s'abattre	jouer
offrir	continuer	ouvrir
aimer	croire	exister
obtenir	refuser	programmer
acheter	rester	s'ouvrir
venir		

Adaptive Language Modeling for Word Prediction

Keith Trnka
University of Delaware
Newark, DE 19716
trnka@cis.udel.edu

Abstract

We present the development and tuning of a topic-adapted language model for word prediction, which improves keystroke savings over a comparable baseline. We outline our plans to develop and integrate style adaptations, building on our experience in topic modeling to dynamically tune the model to both topically and stylistically relevant texts.

1 Introduction

People who use Augmentative and Alternative Communication (AAC) devices communicate slowly, often below 10 words per minute (wpm) compared to 150 wpm or higher for speech (Newell et al., 1998). AAC devices are highly specialized keyboards with speech synthesis, typically providing single-button input for common words or phrases, but requiring a user to type letter-by-letter for other words, called fringe vocabulary. Many commercial systems (e.g., PRC's ECO) and researchers (Li and Hirst, 2005; Trnka et al., 2006; Wandmacher and Antoine, 2007; Matiasek and Baroni, 2003) have leveraged word prediction to help speed AAC communication rate. While the user is typing an utterance letter-by-letter, the system continuously provides potential completions of the current word to the user, which the user may select. The list of predicted words is generated using a language model.

At best, modern devices utilize a trigram model and very basic recency promotion. However, one of the lamented weaknesses of ngram models is their sensitivity to the training data. They require substantial training data to be accurate, and increasingly more data as more of the context is utilized. For example, Lesher et al. (1999) demonstrate that bigram and trigram models for word prediction are not saturated even when trained on 3 million words, in contrast to a unigram model. In addition to the problem of needing substantial amounts of training text to build a reasonable model, ngrams are sensitive to the difference between training and testing/user texts. An ngram model trained on text of a different topic and/or style may perform very poorly compared to a model trained and tested on similar text. Trnka and McCoy (2007) and Wandmacher and Antoine (2006) have demonstrated the domain sensitivity of ngram models for word prediction.

The problem of utilizing ngram models for conversational AAC usage is that no substantial corpora of AAC text are available (much less conversational AAC text). The most similar available corpora are spoken language, but are typically much smaller than written corpora. The problem of corpora for AAC is that similarity and availability are inversely related, illustrated in Figure 1. At one extreme, a very large amount of formal written English is available, however, it is very dissimilar from conversational AAC text, making it less useful for word prediction. At the other extreme, logged text from the current conversation of the AAC user is the most highly related text, but it is extremely sparse. While this trend is demonstrated with a variety of language modeling applications, the problem is more severe for AAC due to the extremely limited availability of AAC text. Even if we train our models on both a large number of general texts in addition to highly related in-domain texts to address the problem, we

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 61–66,
Columbus, June 2008.

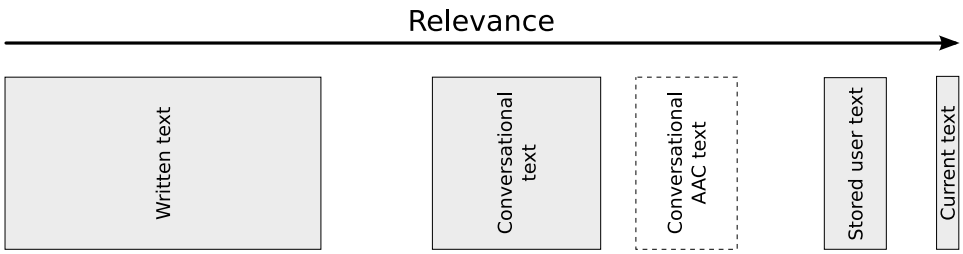


Figure 1: The most relevant text available is often the smallest, while the largest corpora are often the least relevant for AAC word prediction. This problem is exaggerated for AAC.

must focus the models on the most relevant texts.

We address the problem of balancing training size and similarity by dynamically adapting the language model to the most topically relevant portions of the training data. We present the results of experimenting with different topic segmentations and relevance scores in order to tune existing methods to topic modeling. Our approach is designed to seamlessly degrade to the baseline model when no relevant topics are found, by interpolating frequencies as well as ensuring that all training documents contribute some non-zero probabilities to the model. We also outline our plans to adapt ngram models to the style of discourse and then combine the topical and stylistic adaptations.

1.1 Evaluating Word Prediction

Word prediction is evaluated in terms of keystroke savings — the percentage of keystrokes saved by taking full advantage of the predictions compared to letter-by-letter entry.

$$KS = \frac{keys_{\textit{letter-by-letter}} - keys_{\textit{with prediction}}}{keys_{\textit{letter-by-letter}}} \times 100\%$$

Keystroke savings is typically measured automatically by simulating a user typing the testing data of a corpus, where any prediction is selected with a single keystroke and a space is automatically entered after selecting a prediction. The results are dependent on the quality of the language model as well as the number of words in the prediction window. We focus on 5-word prediction windows. Many commercial devices provide optimized input for the most common words (called core vocabulary) and offer word prediction for all other words (fringe vocabulary). Therefore, we limit our evaluation to fringe words only, based on a core vocabulary list from conversations of young adults.

We focus our training and testing on Switchboard, which we feel is similar to conversational AAC text. Our overall evaluation varies the training data from Switchboard training to training on out-of-domain data to estimate the effects of topic modeling in real-world usage.

2 Topic Modeling

Topic models are language models that dynamically adapt to testing data, focusing on the most related topics in the training data. It can be viewed as a two stage process: 1) identifying the relevant topics by scoring and 2) tuning the language model based on relevant topics. Various other implementations of topic adaptation have been successful in word prediction (Li and Hirst, 2005; Wandmacher and Antoine, 2007) and speech recognition (Bellegarda, 2000; Mahajan et al., 1999; Seymore and Rosenfeld, 1997). The main difference of the topic modeling approach compared to Latent Semantic Analysis (LSA) models (Bellegarda, 2000) and trigger pair models (Lau et al., 1993; Matiasek and Baroni, 2003) is that topic models perform the majority of generalization about topic relatedness at testing time rather than training time, which potentially allows user text to be added to the training data seamlessly.

Topic modeling follows the framework below

$$P_{\textit{topic}}(w \mid h) = \sum_{t \in \textit{topics}} P(t \mid h) * P(w \mid h, t)$$

where w is the word being predicted/estimated, h represents all of the document seen so far, and t represents a single topic. The linear combination for topic modeling shows the three main areas of variation in topic modeling. The posterior probability,

$P(w \mid h, t)$ represents the sort of model we have; how topic will affect the adapted language model in the end. The prior, $P(t \mid h)$, represents the way topic is identified. Finally, the meaning of $t \in topics$, requires explanation — what is a topic?

2.1 Posterior Probability — Topic Application

The topic modeling approach complicates the estimation of probabilities from a corpus because the additional conditioning information in the posterior probability $P(w \mid h, t)$ worsens the data sparseness problem. This section will present our experience in lessening the data sparseness problem in the posterior, using examples on trigram models.

The **posterior probability** requires more data than a typical ngram model, potentially causing **data sparseness** problems. We have explored the possibility of estimating it by geometrically combining a topic-adapted unigram model (i.e., $P(w \mid t)$) with a context-adapted trigram model (i.e., $P(w \mid w_{-1}, w_{-2})$), compared to straightforward measurement ($P(w \mid w_{-1}, w_{-2}, t)$). Although the first approach avoids the additional data sparseness, it makes an assumption that the topic of discourse only affects the vocabulary usage. Bellegarda (2000) used this approach for LSA-adapted modeling, however, we found this approach to be inferior to direct estimation of the posterior probability for word prediction (Trnka et al., 2006). Part of the reason for the lesser benefit is that the overall model is only affected slightly by topic adaptations due to the tuned exponential weight of 0.05 on the topic-adapted unigram model. We extended previous research by forcing trigram predictions to occur over bigrams and so on (rather than backoff) and using the topic-adapted model for re-ranking within each set of predictions, but found that the forced ordering of the ngram components was overly detrimental to keystroke savings.

Backoff models for topic modeling can be constructed either before or after the linear interpolation. If the backoff is performed after interpolation, we must also choose whether smoothing (a prerequisite for backoff) is performed before or after the interpolation. If we smooth before the interpolation, then the frequencies will be overly discounted, because the smoothing method is operating on a small fraction of the training data, which will reduce the benefit of higher-order ngrams in the overall model. Also, if we combine probability distributions from each topic, the combination approach may have difficulties with topics of varying size. We address these issues by instead combining frequencies and performing smoothing and backoff after the combination, similar to Adda et al. (1999), although they used corpus-sized topics. The advantage of this approach is that the held-out probability for each distribution is appropriate for the training data, because the smoothing takes place knowing the number of words that occurred in the whole corpus, rather than for each small segment. This is especially important when dealing with small and different sized topics.

The **linear interpolation affects smoothing methods negatively** — because the weights are less than one, the combination decreases the total sum of each conditional distribution. This will cause smoothing methods to underestimate the reliability of the models, because smoothing methods estimate the reliability of a distribution based on the absolute number of occurrences. To correct this, after interpolating the frequencies we found it useful to scale the distribution back to its original sum. The scaling approach improved keystroke savings by 0.2%–0.4% for window size 2–10 and decreased savings by 0.1% for window size 1. Because most AAC systems provide 5–7 predictions, we use this approach. Also, because some smoothing methods operate on frequencies, but the combination model produces real-valued weights for each word, we found it necessary to bucket the combined frequencies to convert them to integers.

Finally, we required an **efficient smoothing** method that could discount each conditional distribution individually to facilitate on-demand smoothing for each conditional distribution, in contrast to a method like Katz' backoff (Katz, 1987) which smoothes an entire ngram model at once. Also, Good-Turing smoothing proved too cumbersome, as we were unable to rely on the ratio between words in given bins and also unable to reliably apply regression. Instead, we used an approximation of Good-Turing smoothing that performed similarly, but allowed for substantial optimization.

2.2 Prior Probability — Topic Identification

The topic modeling approach uses the current testing document to tune the language model to the most relevant training data. The benefit of adaptation is dependent on the quality of the similarity scores. We will first present our representation of the current document, which is compared to unigram models of each topic using a similarity function. We determine the weight of each word in the current document using frequency, recency, and topical salience.

The **recency of use** of a word contributes to the relevance of the word. If a word was used somewhat recently, we would expect to see the word again. We follow Bellegarda (2000) in using an exponentially decayed cache with weight of 0.95 to model this effect of recency on importance at the current position in the document. The weight of 0.95 represents a preservation in topic, but with a decay for very stale words, whereas a weight of 1 turns the exponential model into a pure frequency model and lower weights represent quick shifts in topic.

The importance of each word occurrence in the current document is a factor of not just its frequency and recency, but also it's **topical salience** — how well the word discriminates between topics. For this reason, we decided to use a technique like Inverse Document Frequency (IDF) to boost the weight of words that occur in only a few documents and depress the weights of words that occur in most documents. However, instead of using IDF to measure topical salience, we use Inverse Topic Frequency (ITF), which is more specifically tailored to topic modeling and the particular kinds of topics used.

We evaluated several **similarity functions** for topic modeling, initially using the cosine measure for similarity scoring and scaling the scores to be a probability distribution, following Florian and Yarowsky (1999). The intuition behind the cosine measure is that the similarity between two distributions of words should be independent of the length of either document. However, researchers have demonstrated that cosine is not the best relevance metric for other applications, so we evaluated two other topical similarity scores: Jacquard's coefficient, which performed better than most other similarity measures in a different task for Lee (1999) and Naïve Bayes, which gave better results than cosine in topic-adapted language models for Seymore and Rosenfeld (1997). We evaluated all three similarity metrics using Switchboard topics as the training data and each of our corpora for testing using cross-validation. We found that cosine is consistently better than both Jacquard's coefficient and Naïve Bayes, across all corpora tested. The differences between cosine and the other methods are statistically significant at $p < 0.001$. It may be possible that the ITF or recency weighting in the cache had a negative interaction with Naïve Bayes; traditionally raw frequencies are used.

We found it useful to **polarize the similarity scores**, following Florian and Yarowsky (1999), who found that transformations on cosine similarity reduced perplexity. We scaled the scores such that the maximum score was one and the minimum score was zero, which improved keystroke savings somewhat. This helps fine-tune topic modeling by further boosting the weights of the most relevant topics and depressing the weights of the less relevant topics.

Smoothing the scores helps prevent some scores from being zero due to lack of word overlap. One of the motivations behind using a linear interpolation of all topics is that the resulting ngram model will have the same coverage of ngrams as a model that isn't adapted by topic. However, the similarity score will be zero when no words overlap between the topic and history. Therefore we decided to experiment with similarity score smoothing, which records the minimum nonzero score and then adds a fraction of that score to all scores, then only apply upscaling, where the maximum is scaled to 1, but the minimum is not scaled to zero. In pilot experiments, we found that smoothing the scores did not affect topic modeling with traditional topic clusters, but gave minor improvements when documents were used as topics.

Stemming is another alternative to improving the similarity scoring. This helps to reduce problems with data sparseness by treating different forms of the same word as topically equivalent. We found that stemming the cache representations was very useful when documents were treated as topics (0.2% increase across window sizes), but detrimental when larger topics were used (0.1–0.2% decrease across window sizes). Therefore, we only use stemming when documents are treated as topics.

2.3 What's in a Topic — Topic Granularity

We adapt a language model to the most relevant *topics* in training text. But what is a topic? Traditionally, document clusters are used for topics, where some researchers use hand-crafted clusters (Trnka et al., 2006; Lesher and Rinkus, 2001) and others use automatic clustering (Florian and Yarowsky, 1999). However, other researchers such as Mahajan et al. (1999) have used each individual document as a topic. On the other end of the spectrum, we can use whole corpora as topics when training on multiple corpora. We call this spectrum of topic definitions *topic granularity*, where manual and automatic document clusters are called *medium-grained* topic modeling. When topics are individual documents, we call the approach *fine-grained* topic modeling. In fine-grained modeling, topics are very specific, such as seasonal clothing in the workplace, compared to a medium topic for clothing. When topics are whole corpora, we call the approach *coarse-grained* topic modeling. Coarse-grained topics model much more high-level topics, such as research or news.

The results of testing on Switchboard across different topic granularities are showin in Table 1. The in-domain test is trained on Switchboard only. Out-of-domain training is performed using all other corpora in our collection (a mix of spoken and written language). Mixed-domain training combines the two data sets. Medium-grained topics are only presented for in-domain training, as human-annotated topics were only available for Switchboard. Stemming was used for fine-grained topics, but similarity score smoothing was not used due to lack of time.

The topic granularity experiment confirms our earlier findings that topic modeling can significantly improve keystroke savings. However, the variation of granularity shows that the size of the topics has a strong effect on keystroke savings. Human annotated topics give the best results, though fine-grained topic modeling gives similar results without the need for annotation, making it applicable to training on not just Switchboard but other corpora as well. The coarse grained topic approach seems to be limited to finding acceptable interpolation weights between very similar and very dissimilar data, but is poor at selecting the most relevant corpora from a collection of very different corpora in the out-of-domain test. Another problem may be that many of the corpora are only homogeneous in style but not topic. We would like to extend our work in topic granularity to testing on other corpora in the future.

3 Future Work – Style and Combination

Topic modeling balances the similarity of the training data against the size by tuning a large training set to the most topically relevant portions. However, keystroke savings is not only affected by the topical similarity of the training data, but also the stylistic similarity. Therefore, we plan to also adapt models to the style of text. Our success in adapting to the topic of conversation leads us to believe that a similar process may be applicable to style modeling — splitting the model into style identification and style application. Because we are primarily interested in syntactic style, we will focus on part of speech as the mechanism for realizing grammatical style. As a pilot experiment, we compared a collection of our technical writings on word prediction with a collection of our research emails on word prediction, finding that we could observe traditional trends in the POS ngram distributions (e.g., more pronouns and phrasal verbs in emails). Therefore, we expect that distributional similarity of POS tags will be useful for style identification. We envision a single style s affecting the likelihood of each part of speech p in a POS ngram model like the one below:

$$P(w \mid w_{-1},w_{-2},s) = \sum_{p \in POS(w)} P(p \mid p_{-1},p_{-2},s) * P(w \mid p)$$

In this reformulation of a POS ngram model, the prior is conditioned on the style and the previous couple tags. We will use the overall framework to combine style identification and modeling:

$$P_{style}(w \mid h) = \sum_{s \in styles} P(s \mid h) * P(w \mid w_{-1},w_{-2},s)$$

The topical and stylistic adaptations can be combined by adding topic modeling into the style model shown above. The POS posterior probability $P(w \mid p)$ can be additionally conditioned on the topic of discourse. Topic identification and the topic summation would be implemented consistently with the standalone topic model. Also, the POS framework

Model type	In-domain	Out-of-domain	Mixed-domain
Trigram baseline	60.35%	53.88%	59.80%
Switchboard topics (medium grained)	61.48% (+1.12%)	–	–
Document as topic (fine grained)	61.42% (+1.07%)	54.90% (+1.02%)	61.17% (+1.37%)
Corpus as topic (coarse grained)	–	52.63% (-1.25%)	60.62% (+0.82%)

Table 1: Keystroke savings across different granularity topics and training domains, tested on Switchboard. Improvement over baseline is shown in parentheses. All differences from baseline are significant at $p < 0.001$

facilitates cache modeling in the posterior, allowing direct adaptation to the current text, but with less sparseness than other context-aware models.

4 Conclusions

We have created a topic adapted language model that utilizes the full training data, but with focused tuning on the most relevant portions. The inclusion of all the training data as well as the usage of frequencies addresses the problem of sparse data in an adaptive model. We have demonstrated that topic modeling can significantly increase keystroke savings for traditional testing as well as testing on text from other domains. We have also addressed the problem of annotated topics through fine-grained modeling and found that it is also a significant improvement over a baseline ngram model. We plan to extend this work to build models that adapt to both topic and style.

Acknowledgments

This work was supported by US Department of Education grant H113G040051. I would like to thank my advisor, Kathy McCoy, for her help as well as the many excellent and thorough reviewers.

References

Gilles Adda, Michèle Jardino, and Jean-Luc Gauvain. 1999. Language modeling for broadcast news transcription. In *Eurospeech*, pages 1759–1762.

Jerome R. Bellegarda. 2000. Large vocabulary speech recognition with multispan language models. *IEEE Transactions on Speech and Audio Processing*, 8(1):76–84.

Radu Florian and David Yarowsky. 1999. Dynamic Nonlocal Language Modeling via Hierarchical Topic-Based Adaptation. In *ACL*, pages 167–174.

Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics Speech and Signal Processing*, 35(3):400–401.

R. Lau, R. Rosenfeld, and S. Roukos. 1993. Trigger-based language models: a maximum entropy approach. In *ICASSP*, volume 2, pages 45–48.

Lillian Lee. 1999. Measures of distributional similarity. In *ACL*, pages 25–32.

Gregory Lesher and Gerard Rinkus. 2001. Domain-specific word prediction for augmentative communication. In *RESNA*, pages 61–63.

Gregory W. Lesher, Bryan J. Moulton, and D. Jeffery Higgonbotham. 1999. Effects of ngram order and training text size on word prediction. In *RESNA*, pages 52–54.

Jianhua Li and Graeme Hirst. 2005. Semantic knowledge in word completion. In *ASSETS*, pages 121–128.

Milind Mahajan, Doug Beeferman, and X. D. Huang. 1999. Improved topic-dependent language modeling using information retrieval techniques. In *ICASSP*, volume 1, pages 541–544.

Johannes Matiasek and Marco Baroni. 2003. Exploiting long distance collocational relations in predictive typing. In *EACL-03 Workshop on Language Modeling for Text Entry*, pages 1–8.

Alan Newell, Stefan Langer, and Marianne Hickey. 1998. The rôle of natural language processing in alternative and augmentative communication. *Natural Language Engineering*, 4(1):1–16.

Kristie Seymore and Ronald Rosenfeld. 1997. Using Story Topics for Language Model Adaptation. In *Eurospeech*, pages 1987–1990.

Keith Trnka and Kathleen F. McCoy. 2007. Corpus Studies in Word Prediction. In *ASSETS*, pages 195–202.

Keith Trnka, Debra Yarrington, Kathleen McCoy, and Christopher Pennington. 2006. Topic Modeling in Fringe Word Prediction for AAC. In *IUI*, pages 276–278.

Tonio Wandmacher and Jean-Yves Antoine. 2006. Training Language Models without Appropriate Language Resources: Experiments with an AAC System for Disabled People. In *LREC*.

T. Wandmacher and J.Y. Antoine. 2007. Methods to integrate a language model with semantic information for a word prediction component. In *EMNLP*, pages 506–513.

A Hierarchical Approach to Encoding Medical Concepts for Clinical Notes

Yitao Zhang
School of Information Technologies
The University of Sydney
NSW 2006, Australia
yitao@it.usyd.edu.au

Abstract

This paper proposes a hierarchical text categorization (TC) approach to encoding free-text clinical notes with ICD-9-CM codes. Preliminary experimental result on the 2007 Computational Medicine Challenge data shows a hierarchical TC system has achieved a micro-averaged F_1 value of 86.6, which is comparable to the performance of state-of-the-art flat classification systems.

1 Introduction

The task of assigning meaningful categories to free text has attracted researchers in the Natural Language Processing (NLP) and Information Retrieval (IR) field for more than 10 years. However, it has only recently emerged as a hot topic in the clinical domain where categories to be assigned are organized in taxonomies which cover common medical concepts and link them together in hierarchies. This paper evaluates the effectiveness of adopting a hierarchical text categorization approach to the 2007 Computational Medicine Challenge which aims to assign appropriate ICD-9-CM codes to free text radiology reports. (Pestian et al., 2007)

The ICD-9-CM [1], which stands for International Classification of Diseases, 9th Revision, Clinical Modification, is an international standard which is used for classifying common medical concepts, such as diseases, symptoms and signs, by hospitals, insurance companies, and other health organizations. The 2007 Computational Medicine Challenge was set in a billing scenario in which hospitals claim reimbursement from health insurance companies based on the ICD-9-CM codes assigned to each patient case. The competition has successfully attracted 44 submissions with a mean micro-averaged F_1 performance of 76.70. (Pestian et al., 2007)

To the best of our knowledge, the systems reported were all adopting a flat classification approach in which a dedicated classifier has been built for every targeted ICD-9-CM code. Each classifier makes a binary decision of True or False according to whether or not a clinical note should be assigned with the targeted ICD-9-CM code. An incoming clinical note has to be tested against all the classifiers before a final coding decision can be made. The response time of a flat approach therefore grows linearly with the number of categories in the taxonomy. Moreover, low-frequency ICD-9-CM codes suffer the data imbalance problem in which positive training instances are overwhelmed by negative ones.

A hierarchical system takes into account relationships among categories. Classifiers are assigned to both leaf and internal nodes of a taxonomy and training instances are distributed among these nodes. When a test instance comes in, a coding decision is made by generating all possible paths (start from the root node of the taxonomy) where classifiers along path return favorable decisions. In other words, a node is visited only if the classifier assigned to its parent returns a True decision. This strategy significantly reduces the average number of classifiers to be used in the test stage when the taxonomy is very large. (Liu et al., 2005; Yang et al., 2003)

[1] see http://www.cdc.gov/nchs/icd9.htm

Proceedings of the ACL-08: HLT Student Research Workshop (Companion Volume), pages 67–72,
Columbus, June 2008.

2 Related Works

Most top systems in the 2007 Computational Medicine Challenge have benefited from incorporating domain knowledge of free-text clinical notes, such as negation, synonymy, and hypernymy, either as hand-crafted rules in a symbolic approach, or as carefully engineered features in a machine-learning component. (Goldstein et al., 2007; Farkas and Szarvas, 2007; Crammer et al., 2007; Aronson et al., 2007; Patrick et al., 2007)

Aronson et al. (2007) used a variant of National Library of Medicine Medical Text Indexer (MTI) which was originally developed for discovering Medical Subject Headings (MeSH) [2] terms for indexing biomedical citations and articles. The output of MTI was converted into ICD-9-CM codes by applying different approaches of mapping discovered Unified Medical Language System (UMLS) [3] concepts into ICD-9-CM codes, such as using synonym and built-in mapping relations in UMLS Metathesaurus. This approach can easily adapt to any subdomain of the UMLS Metathesaurus since it only requires very little examples for tuning purposes. However, MTI performed slightly behind an SVM system with only bag-of-words features, which suggests the difficulty of optimizing a general purpose system without any statistical learning on the targeted corpus. By stacking MTI, SVM, KNN and a simple pattern matching system together, a final F_1 score of 85 was reported on the official test set.

Farkas and Szarvas (2007) automatically translate definitions of the ICD-9-CM into rules of a symbolic system. Decision tree was then used to model the disagreement between the prediction of the system and the gold-standard annotation of the training data set. This has improved the performance of the system to a F_1 value of 89. Goldstein et al. (2007) also reported that a rule-based system enhanced by negation, synonymy, and uncertainty information, has outperformed machine learning models which only use n-gram features. The rules were manually tuned for every ICD-9-CM code found in the challenge training data set and therefore suffer the scaling up problem.

On the other hand, researchers tried to encode domain knowledge into machine learning systems by developing more sophisticated feature types. Patrick et al. (2007) developed a variety of new feature types to model human coder's expertise, such as negation and code overlaps. Different combination of feature types were tested for each individual ICD-9-CM code and the best combination was used in the final system. Crammer et al. (2007) also used a rich feature set in their MIRA system which is an online learning algorithm.

[2] http://www.nlm.nih.gov/mesh/

[3] http://www.nlm.nih.gov/research/umls/

Total radiology records	1,954
Total tokens	51,940
Total ICD-9-Codes	45
Total code instances	2,423

Table 1: Statistics of the data set

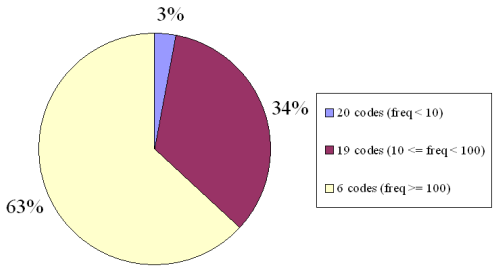


Figure 1: Distribution of ICD-9-CM codes in the challenge data set.

3 The Corpus

The corpus used in this study is the official data set of the 2007 Computational Medicine Challenge. The challenge corpus consists of 1,954 radiology reports from the Cincinnati Children's Hospital Medical Center and was divided into a training set with 978 records, and a test set with 976 records. The statistics of the corpus is shown in Table 1.

Each radiology record in the corpus has two sections: 'Clinical History' which is provided by an ordering physician before a radiological procedure, and 'Impression' which is reported by a radiologist after the procedure. A typical radiology report is shown below:

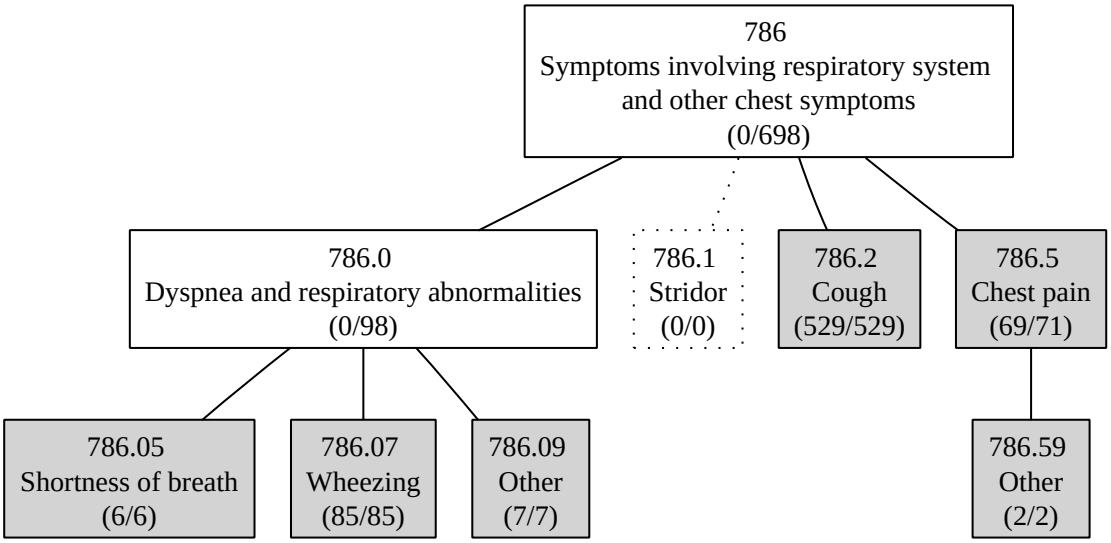


Figure 2: A part of the ICD-9-CM taxonomy: the tree covers symptoms involving respiratory system and other chest symptoms. There are two figures shown in each node: the first figure is the number of positive instances assigned to the current node, and the next figure shows the number of all the instances in its subtree.

Clinical history

Persistent cough, no fever.

Impression

Retained secretions vs atelectasis in the right lower lobe. No infiltrates to support pneumonia

Three different institutions were invited to assign ICD-9-CM codes to the corpus. The majority code with at least two votes from the three annotators was considered as the gold-standard code for the record. Moreover, a clinical record can be assigned with multiple ICD-9-CM codes at a time.

The general guideline of assigning ICD-9-CM codes includes two important rules:

- If there is a definite diagnosis in text, the diagnosis should be coded and all symptom and sign codes should be ignored.

- If the diagnosis is undecided, or there is no diagnosis found, the symptoms and signs should be coded rather than the uncertain diagnosis.

According to the guideline, the above radiology record should be assigned with only a 'Cough' code because 'Atelectasis' and 'Pneumonia' are not certain, and 'Fever' has been negated.

There are 45 ICD-9-CM codes found in the corpus and their distribution is imbalanced. Figure 1 shows a pie chart of three types of the ICD-9-CM codes found in the corpus and their accumulated category frequencies. The 20 low-frequency (less than 10 occurrences) codes account for only 3% of the total code occurrence in the challenge data set. There are 19 codes with a frequency between 10 and 100 and altogether they account for 34% total code occurrence. Finally, the most frequent six codes account for over 60% of total code instances.

4 Hierarchical Text Categorization Framework

In a hierarchical text categorization system, categories are linked together and classifiers are assigned to each node in the taxonomy. In the training stage, instances are distributed to their corresponding nodes. For instance, Figure 2 shows a populated subtree of ICD-9-CM code '786' which covers concepts involving respiratory system and other chest symptoms. Nodes in grey box such as 786.2 and 786.5 are among 45 gold-standard codes found in the challenge data set. Nodes in white box such as 786 and 786.0 are internal nodes which have non-

empty subtrees. For instance, the numbers (0, 698) of '786' suggest that the node is assigned with zero instances for training while there are 698 positive instances assigned to nodes in its subtree. The node '786.1' is in dotted box because there is no instance assigned to it, nor any of its subtrees. In the experiment, all nodes (such as '786.1') with empty instance in its subtree were removed from the training and testing stage.

When training a classifier for a node A in the tree, all the instances in the subtree rooted in the parent of A become the only source of training instances. For instance, code '786.0' in Figure 2 uses all the 698 instances rooted in node '786' as the full training data set. The 98 instances rooted in node '786.0' itself are the positive instances while the remaining 600 instances in the tree as the negative ones. This hierarchical approach of distributing training instances can reduce the size of training data set for most classifiers and minimize the data imbalance problem for low-frequency codes in the taxonomy.

In the test stage, the system starts from the root of the ICD-9-CM taxonomy and evaluates an incoming clinical note against classifiers assigned to its children nodes. The system will then visit every child node which returns a positive classification result. The process repeats recursively until a possible path ends by reaching a node that returns a negative classification result. This strategy enables the sytem to assign multiple codes to a clinical note by visiting different paths in the ICD-9-CM taxonomy simultaneously.

5 Methods and Experiments

5.1 Experiment Settings

In this study, Support Vector Machines (SVM) was used for both flat and hierarchical text categorization. The LibSVM (Chang and Lin, 2001) package was used with a linear kernel.

5.1.1 Hierarchical TC

A tree of ICD-9-CM taxonomy was constructed by enquiring the UMLS Metathesaurus. During each iteration of 10-fold cross-validation experiment, the training instances were assigned to the ICD-9-CM tree and all nodes assigned with zero training instance in its subtree were removed from the tree. This ended with an ICD-9-CM tree with around 100 nodes for each training and test iteration.

Nodes in the tree were uniquely identified by their concept id (CUI) found in the UMLS Metathesaurus. However, two ICD-9-CM codes ('599.0' and 'V13.02') were found to share the same CUI in the UMLS Metathesaurus. As a result, 44 unique UMLS CUIs were used as the gold-standard codes in the experiment for the original 45 ICD-9-CM codes.

In the test stage, the hierarchical system returns the terminal nodes of the predicted path. Moreover, if the terminal ends in an internal code which is not one of the 44 gold-standard UMLS CUI found in the training corpus, the system should ignore the whole path.

5.1.2 Flat TC

In a flat text categorization setting, 44 classifiers were created for each UMLS Metathesaurus CUI found in the corpus. Each classifier makes a binary decision of 'Yes' or 'No' to a clinical record according to whether or not it should be assigned with the current code.

5.2 Preprocessing

The corpus was first submitted to the GENIA tagger (Tsuruoka et al., 2005) for part-of-speech tagging and shallow syntactic analysis. The result was used by the negation finding module and all the identified negated terms were removed from the corpus. The cleaned text was used by the MetaMap (Aronson, 2001) for identifying possible medical concepts in text. The MetaMap software is configured to return only concepts of ICD-9-CM and SNOMED CT which is another comprehensive medical ontology widely used for mapping concepts in free-text clinical notes.

5.3 Evaluation

The main evaluation metric used in the experiment is the micro-averaged F_1 which is defined as the harmonic mean between $Precision$ and $Recall$:

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

where

$$Precision = \frac{\sum_i TP(Code_i)}{\sum_i TP(Code_i) + \sum_i FP(Code_i)}$$

$$Recall = \frac{\sum_i TP(Code_i)}{\sum_i TP(Code_i) + \sum_i FN(Code_i)}$$

In the above equation, $TP(Code_i)$, $FP(Code_i)$, and $FN(Code_i)$ are the numbers of true positives, false positives, and false negatvies for the ith code. The micro-averaged F_1 considers every single coding decision equally important and is therefore dominant by the performance on frequent codes in data. Moreover, a hierarchical micro-averaged $F_1^{(hierarchical)}$ is also introduced by adding all ancestors of the current gold-standard code into calculation. The $F_1^{(hierarchical)}$ value helps to evaluate how accurate a system predicts in terms of the gold-standard path in the ICD-9-CM tree.

5.4 Features

The feature set is descibed in Table 2.

- Bag-of-words

  Both unigram (F1) and bigram (F2) were used.

- Negation and Bag-of-concepts

  An algorithm similar to NegEx (Chapman et al., 2001) was used to find negations in text. A small set of 35 negation keywords, such as 'no', 'without', and 'no further', was compiled to trigger the finding of the negated phrases in text based on the shallow syntactic analysis returned by GENIA tagger. After removing negated phrases in text, MetaMap was used to find medical concepts in text as new features in a bag-of-concepts manner (F3 and F4).

Different combination of feature types (F5, F6, and F7) were also used in the experiment. Information gain was used to rank the features and the feature cut-off threshold was set to 4, 000.

6 Result and Discussion

The 10-fold cross-validation technique was used in the experiments. The 1,954 radiology reports were randomly divided into ten folds. In each iteration of the experiment, one fold of data was used as the test set and the other nine folds as the training set.

The experimental results are shown in Table 2. The flat TC system has achieved higher F_1 scores than a hierarchical TC system in all experimental settings. However, paired t-test suggests the differences are not statistically significant at a ($p < 0.05$) level in most cases. This suggests the potential of adopting a hierarchical TC approach in the task. The effectiveness of the system is not sacrificed while the system now has the potential to scale up to much larger problems.

Similarly, the hierarchical TC system has better $F_1^{hierarchical}$ scores than the flat TC system while this difference is still not statistically significant at a ($p < 0.05$) level in most cases. This is partly due to the current strategy of not allowing unknown ICD-9-CM codes to be assigned in the system. As a result, many originally predicted internal nodes were removed in a hierarchical TC system.

Both the flat and hierarchical systems using bag-of-words feature set F1 have achieved a F_1 score above 0.85. Adding bigram features into F2 has shown minimum impact on the performance of both systems. Using a bag-of-concepts strategy in F3 and F4 has lowered the performance of the system. However, adding F3 and F4 into bag-of-words feature set has improved the performance of both systems. Finally, the best performance were reported on using feature set F5 which combines unigram and ICD-9-CM concepts returned by MetaMap software on the preprocessed text where negated terms were removed.

7 Conclusion and Future Work

Compared to a flat classification approach, a hierarchical framework is able to exploit relationships among categories to be assigned and easily adapts to much larger text categorization problems where real-time response is needed. This study has proposed a hierarchical text categorization approach to the task of encoding clinical notes with ICD-9-CM codes. The preliminary experiment shows that a hierarchical text categorization system has achieved a performance comparable to other state-of-the-art flat classification systems.

Future work includes developing more sophisticated features, such as synonym and phrase-level paraphrasing and entailment, to encode the knowl-

	Feature Description	Flat TC		Hierarchical TC	
		F_1	$F_1^{(hierarchical)}$	F_1	$F_1^{(hierarchical)}$
F1	Unigram	85.90 ± 2.00	89.50 ± 1.51	85.52 ± 1.30	90.49 ± 1.13
F2	Unigram, Bigram	85.99 ± 2.17	89.65 ± 1.70	85.27 ± 1.32	90.69 ± 1.20
F3	ICD-9-CM concepts on no negation text	81.96 ± 1.44	85.39 ± 1.47	81.45 ± 1.79	86.89 ± 1.65
F4	SNOMED CT concepts on no negation text	84.97 ± 1.55	89.00 ± 1.04	84.77 ± 1.04	89.82 ± 0.97
F5	F1 + F3	87.09 ± 1.70	90.26 ± 1.33	86.58 ± 1.30	91.08 ± 0.95
F6	F1 + F4	86.56 ± 1.69	89.99 ± 1.34	86.10 ± 1.80	90.70 ± 1.58
F7	F1 + F3 + F4	86.83 ± 1.34	90.23 ± 1.17	86.57 ± 1.28	91.06 ± 1.10

Table 2: 10-fold cross-validation experimental results

edge of human experts. How to manage a rich feature set in a hierarchical TC setting would be another big challenge. Moreover, this work did not use any thresholding tuning technique in the training stage. Therefore, a thorough study on the effectiveness of threshold tuning in the task is required.

Acknowledgments

I would like to thank Prof. Jon Patrick for his support and supervision of my research, and Mr. Yefeng Wang for providing his codes on negation finding. I also want to thank all the anonymous reviewers for their invaluable inputs to my research.

References

A.R. Aronson, O. Bodenreider, D. Demner-Fushman, K.W. Fung, V.K. Lee, J.G. Mork, A. Névéol, L. Peters, and W.J. Rogers. 2007. From Indexing the Biomedical Literature to Coding Clinical Text: Experience with MTI and Machine Learning Approaches. *Proceedings of the Workshop on BioNLP 2007*, pages 105–112.

A.R. Aronson. 2001. Effective Mapping of Biomedical Text to the UMLS Metathesaurus: the MetaMap Program. *Proc AMIA Symp*, pages 17–21.

C. C. Chang and C. J. Lin, 2001. *LIBSVM: a Library for Support Vector Machines*. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

W.W. Chapman, W. Bridewell, P. Hanbury, G.F. Cooper, and B.G. Buchanan. 2001. A Simple Algorithm for Identifying Negated Findings and Diseases in Discharge Summaries. *Journal of Biomedical Informatics*, 34(5):301–310.

K. Crammer, M. Dredze, K. Ganchev, P.P. Talukdar, and S. Carroll. 2007. Automatic Code Assignment to Medical Text. *Proceedings of the Workshop on BioNLP 2007*, pages 129–136.

R. Farkas and G. Szarvas. 2007. Automatic Construction of Rule-based ICD-9-CM Coding Systems. *The Second International Symposium on Languages in Biology and Medicine*.

I. Goldstein, A. Arzumtsyan, and Ö. Uzuner. 2007. Three Approaches to Automatic Assignment of ICD-9-CM Codes to Radiology Reports. *AMIA Annu Symp Proc*.

T.Y. Liu, Y. Yang, H. Wan, H.J. Zeng, Z. Chen, and W.Y. Ma. 2005. Support Vector Machines Classification with a Very Large-scale Taxonomy. *SIGKDD Explorations, Special Issue on Text Mining and Natural Language Processing*, 7(1):36–43.

J. Patrick, Y. Zhang, and Y. Wang. 2007. Evaluating Feature Types for Encoding Clinical Notes. *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pages 218–225.

J.P. Pestian, C. Brew, P. Matykiewicz, DJ Hovermale, N. Johnson, K.B. Cohen, and W. Duch. 2007. A Shared Task Involving Multi-label Classification of Clinical Free Text. *Proceedings of the Workshop on BioNLP 2007*, pages 97–104.

Y. Tsuruoka, Y. Tateishi, J.D. Kim, T. Ohta, J. McNaught, S. Ananiadou, and J. Tsujii. 2005. Developing a Robust Part-of-Speech Tagger for Biomedical Text. In *Advances in Informatics - 10th Panhellenic Conference on Informatics*, pages 382–392.

Y. Yang, J. Zhang, and B. Kisiel. 2003. A Scalability Analysis of Classifiers in Text Categorization. *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 96–103.

Author Index