NAACL HLT 2009

Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics

Proceedings of the Student Research Workshop and Doctoral Consortium

Anoop Sarkar, Carolyn Rose, Svetlana Stoyanchev, Ulrich Germann and Chirag Shah
Chairs

June 1, 2009
Boulder, Colorado

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

We gratefully acknowledge financial support from the U.S. 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

ISBN 978-1-932432-42-8

Preface

Welcome to the NAACL HLT 2009 Student Research Workshop and Doctoral Consortium! We are pleased to continue this established tradition at ACL conferences.

The Student Research Workshop and Doctoral Consortium provides a venue for student researchers in Computational Linguistics, Natural Language Processing, and Human Language Technologies to present their work and receive feedback from the community. Unlike regular conference sessions, the workshop welcomes work in progress. In the tradition of previous doctoral consortia, we recruited expert panelists to provide a brief commentary on each oral presentation. We hope that as a result of this workshop, the student participants are able to obtain exposure to the NAACL community and that it helps with their future careers in this field.

This year we received 29 submissions from 11 countries. With a strong program committee of 86 reviewers, from advanced graduate students to senior experts in their field, we were able to keep the individual reviewing load light and assign to each paper a team of three to six reviewers. Most papers were reviewed by two student reviewers and two established researchers.

We would like to thank the reviewers for understanding the spirit of the workshop and giving careful and constructive reviews. We hope their comments will be helpful to all the students who submitted their work.

A grant from the U.S. National Science Foundation enables us to provide financial support to all presenters to assist them in their travel to and attendance of the conference. We gratefully acknowledge this generous contribution.

Finally, we would also like to thank the general chair of NAACL HLT 2009, Mari Ostendorf, the program committee chairs, Michael Collins, Lucy Vanderwende, Doug Oard, and Shri Narayanan, the publicity chairs Matthew Stone, Gokhan Tur, and Diana Inkpen, the publications chairs, Christy Doran and Eric Ringger, the local arrangements chairs, James Martin and Martha Palmer, and the ACL Business Manager, Priscilla Rasmussen, for all the support they have provided in the organization of this workshop.

The faculty advisors and co-chairs of the NAACL HLT 2009
Student Research Workshop and Doctoral Consortium:

Carolyn Penstein Rosé
Anoop Sarkar

Svetlana Stoyanchev
Ulrich Germann
Chirag Shah

Co-chairs:

Ulrich Germann, University of Toronto, Canada
Chirag Shah, University of North Carolina, USA
Svetlana Stoyanchev, Stony Brook University, USA

Faculty Advisors:

Carolyn Penstein Rosé, Carnegie Mellon University, USA
Anoop Sarkar, Simon Fraser University, Canada

Program Committee:

Afra Alishahi
Nguyen Bach
Niranjan Balasubramanian
Satanjeev Banerjee
Regina Barzilay
Shane Bergsma
Alan Black
Dan Bohus
Sarah Borys
Chris Callison-Burch
Claire Cardie
Marine Carpuat
Colin Cherry
Yejin Choi
Paul Cook
Steve DeNeefe
Fernando Diaz
Gregory Duck
Kevin Duh
Jason Eisner
Afsaneh Fazly
Jenny Finkel
Victoria Fossum
George Foster
Yanfen Hao
Sanjika Hewavitharana
Derrick Higgins
Silja Hildebrand
Graeme Hirst
Pierre Isabelle
Howard Johnson
Pallika Kanani
Weimao Ke
Diane Kelly
Kevin Knight
Philipp Koehn
Greg Kondrak
Roland Kuhn
Shankar Kumar
Giridhar Kumaran
Philippe Langlais
Brian Langner
Gregor Leusch
Adam Lopez
Annie Louis
Daniel Marcu
Jonathan May
David McClosky
David Mimno
Saif Mohammad
Antonio Moreno
Javed Mostafa
Cristina Mota
Smaranda Muresan
Ani Nenkova
Thuylinh Nguyen
Oana Nicolov
Christopher Parisien
Joana Paulo Pardal
Ted Pedersen
Gerald Penn
Hema Raghavan
Ricardo Ribeiro
Jason Riesa
Ellen Riloff
Mihai Rotaru
Alex Rudnicky
Frank Rudzicz
Narges Sharif Razavian
Michel Simard
David Smith
Yaxiao Song
Richard Sproat
Amanda Stent
Veselin Stoyanov
Joel Tetrault
Dilek Tur
Suzan Verberne
Stephan Vogel
Qin Iris Wang
Nick Webb
Ryen White
Fan Yang
Su-Youn Yoon
Klaus Zechner
Xiaodan Zhu

Table of Contents

Conference Program

Monday, June 1, 2009

Morning Session

10:40–11:10 *Classifier Combination Techniques Applied to Coreference Resolution*
Smita Vemulapalli, Xiaoqiang Luo, John F. Pitrelli and Imed Zitouni

11:15–11:45 *Solving the "Who's Mark Johnson Puzzle": Information Extraction Based Cross Document Coreference*
Jian Huang, Sarah M. Taylor, Jonathan L. Smith, Konstantinos A. Fotiadis and C. Lee Giles

11:50–12:20 *Exploring Topic Continuation Follow-up Questions using Machine Learning*
Manuel Kirschner and Raffaella Bernardi

First Afternoon Session

2:00–2:30 *Sentence Realisation from Bag of Words with Dependency Constraints*
Karthik Gali and Sriram Venkatapathy

2:35–3:05 *Using Language Modeling to Select Useful Annotation Data*
Dmitriy Dligach and Martha Palmer

Second Afternoon Session

4:00–4:30 *Pronunciation Modeling in Spelling Correction for Writers of English as a Foreign Language*
Adriane Boyd

4:35–5:05 *Building a Semantic Lexicon of English Nouns via Bootstrapping*
Ting Qian, Benjamin Van Durme and Lenhart Schubert

5:10–5:40 *Multiple Word Alignment with Profile Hidden Markov Models*
Aditya Bhargava and Grzegorz Kondrak

Monday, June 1, 2009 (continued)

Poster Session (6:30–9:30)

Using Emotion to Gain Rapport in a Spoken Dialog System
Jaime Acosta

Interactive Annotation Learning with Indirect Feature Voting
Shilpa Arora and Eric Nyberg

Loss-Sensitive Discriminative Training of Machine Transliteration Models
Kedar Bellare, Koby Crammer and Dayne Freitag

Syntactic Tree-based Relation Extraction Using a Generalization of Collins and Duffy Convolution Tree Kernel
Mahdy Khayyamian, Seyed Abolghasem Mirroshandel and Hassan Abolhassani

Towards Building a Competitive Opinion Summarization System: Challenges and Keys
Elena Lloret, Alexandra Balahur, Manuel Palomar and Andrés Montoyo

Domain-Independent Shallow Sentence Ordering
Thade Nahnsen

Towards Unsupervised Recognition of Dialogue Acts
Nicole Novielli and Carlo Strapparava

Modeling Letter-to-Phoneme Conversion as a Phrase Based Statistical Machine Translation Problem with Minimum Error Rate Training
Taraka Rama, Anil Kumar Singh and Sudheer Kolachina

Disambiguation of Preposition Sense Using Linguistically Motivated Features
Stephen Tratz and Dirk Hovy

All papers presented in the morning and afternoon sessions will also be shown as posters.

Classifier Combination Techniques Applied to Coreference Resolution

Smita Vemulapalli[1], **Xiaoqiang Luo**[2], **John F. Pitrelli**[2] **and Imed Zitouni**[2]

[1]Center for Signal and Image Processing (CSIP)
School of ECE, Georgia Institute of Technology
Atlanta, GA 30332, USA
smita@ece.gatech.edu

[2]IBM T. J. Watson Research Center
1101 Kitchawan Road
Yorktown Heights, NY 10598, USA
{xiaoluo,pitrelli,izitouni}@us.ibm.com

Abstract

This paper examines the applicability of classifier combination approaches such as bagging and boosting for coreference resolution. To the best of our knowledge, this is the first effort that utilizes such techniques for coreference resolution. In this paper, we provide experimental evidence which indicates that the accuracy of the coreference engine can potentially be increased by use of bagging and boosting methods, without any additional features or training data. We implement and evaluate combination techniques at the mention, entity and document level, and also address issues like entity alignment, that are specific to coreference resolution.

1 Introduction

Coreference resolution is the task of partitioning a set of mentions (*i.e.* person, organization and location) into entities. A *mention* is an instance of textual reference to an object, which can be either named (*e.g.* Barack Obama), nominal (*e.g.* the president) or pronominal (*e.g.* he, his, it). An *entity* is an aggregate of all the mentions (of any level) which refer to one conceptual entity. For example, in the following sentence:

John said Mary was his sister.

there are four mentions: John, Mary, his, and sister.

John and his belong to the one entity since they refer to the same person; Mary and sister both refer to another person entity. Furthermore, John and Mary are *named* mentions, sister is a *nominal* mention and his is a *pronominal* mention.

In this paper, we present a potential approach for improving the performance of coreference resolution by using classifier combination techniques such as bagging and boosting. To the best of our knowledge, this is the first effort that utilizes classifier combination for improving coreference resolution.

Combination methods have been applied to many problems in natural-language processing (NLP). Examples include the ROVER system (Fiscus, 1997) for speech recognition, the Multi-Engine Machine Translation (MEMT) system (Jayaraman and Lavie, 2005), and part-of-speech tagging (Brill and Wu, 1998; Halteren *et al.*, 2001). Most of these techniques have shown a considerable improvement over the performance of a single classifier and, therefore, lead us to consider implementing such a multiple-classifier system for coreference resolution as well.

Using classifier combination techniques one can potentially achieve a classification accuracy that is superior to that of the single best classifier. This is based on the assumption that the errors made by each of the classifiers are not identical, and therefore if we intelligently combine multiple classifier outputs, we may be able to correct some of these errors.

The main contributions of this paper are:

- *Demonstrating the potential for improvement in the baseline* – By implementing a system that behaves like an oracle, we have shown that the output of the combination of multiple classifiers has the potential to be significantly higher in accuracy than any of the individual classifiers.
- *Adapting traditional bagging techniques* – Multiple classifiers, generated using bagging techniques, were combined using an entity-level sum

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 1–6,
Boulder, Colorado, June 2009.

rule and mention-level majority voting.

- *Implementing a document-level boosting algorithm* – A boosting algorithm was implemented in which a coreference resolution classifier was iteratively trained using a re-weighted training set, where the reweighting was done at the document level.
- *Addressing the problem of entity alignment* – In order to apply combination techniques to multiple classifiers, we need to address entity-alignment issues, explained later in this paper.

The baseline coreference system we use is similar to the one described by Luo *et al.* (Luo *et al.*, 2004). In such a system, mentions are processed sequentially, and at each step, a mention is either linked to one of existing entities, or used to create a new entity. At the end of this process, each possible partition of the mentions corresponds to a unique sequence of link or creation actions, each of which is scored by a statistical model. The one with the highest score is output as the final coreference result.

2 Classifier Combination Techniques

2.1 Bagging

One way to obtain multiple classifiers is via bagging or bootstrap aggregating (Breiman, 1996). These classifiers, obtained using randomly-sampled training sets, may be combined to improve classification.

We generated several classifiers by two techniques. In the first technique, we randomly sample the set of documents (training set) to generate a few classifiers. In the second technique, we need to reduce the feature set and this is not done in a random fashion. Instead, we use our understanding of the individual features and also their relation to other features to decide which features may be dropped.

2.2 Oracle

In this paper, we refer to an *oracle* system which uses knowledge of the truth. Here, truth, called the *gold standard* henceforth, refers to mention detection and coreference resolution done by a human for each document. It is possible that the gold standard may have errors and is not perfect truth, but, as in most NLP systems, it is considered the reference for evaluating computer-based coreference resolution.

To understand the oracle, consider an example in which the outputs of two classifiers for the same input document are C_1 and C_2, as shown in Figure 1.

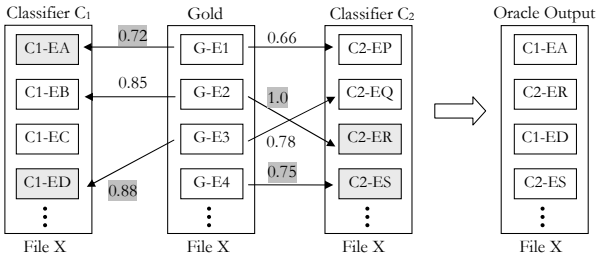

Figure 1: Working of the oracle

The number of entities in C_1 and C_2 may not be the same and even in cases where they are, the number of mentions in corresponding entities may not be the same. In fact, even finding the corresponding entity in the other classifier output or in the gold standard output G is not a trivial problem and requires us to be able to align any two classifier outputs.

The alignment between any two coreference labelings, say C_1 and G, for a document is the best one-to-one map (Luo, 2005) between the entities of C_1 and G. To align the entities of C_1 with those of G, under the assumption that an entity in C_1 may be aligned with at most only one entity in G and vice versa, we need to generate a bipartite graph between the entities of C_1 and G. Now the alignment task is a maximum bipartite matching problem. This is solved by using the Kuhn-Munkres algorithm (Kuhn, 1955; Munkres, 1957). The weights of the edges of the graph are entity-level alignment measures. The metric we use is a relative measure of the similarity between the two entities. To compute the similarity metric ϕ (Luo, 2005) for the entity pair (R, S), we use the formula shown in Equation 1, where $(\cap)$ represents the commonality with attribute-weighted partial scores. Attributes are things such as (ACE) entity type, subtype, entity class, etc.

$$\phi(R, S) = \frac{2\,|R \cap S|}{|R| + |S|} \tag{1}$$

The oracle output is a combination of the entities in C_1 and C_2 with the highest entity-pair alignment measures with the entities in G.[1] We can see in Figure 1 that the entity G-E1 is aligned with entities C1-EA and C2-EP. We pick the entity with the highest entity-pair alignment measure (highlighted in gray) which, in this case, is C1-EA. This is repeated for

[1] A mention may be repeated across multiple output entities, which is not an unwarranted advantage as the scorer insists on one-to-one entity alignment. So if there are two entities containing mention A, at most one mention A is credited and the other will hurt the score.

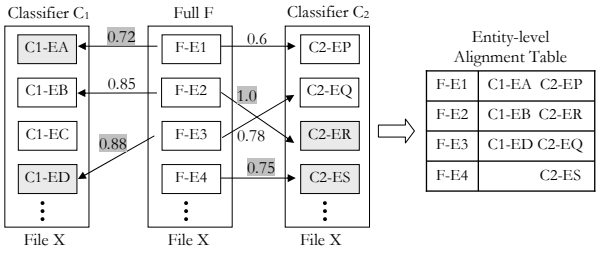

Figure 2: Entity alignment between classifier outputs

every entity in G. The oracle output can be seen in the right-hand side of Figure 1. This technique can be scaled up to work for any number of classifiers.

2.3 Preliminary Combination Approaches

Imitating the oracle. Making use of the existing framework of the oracle, we implement a combination technique that imitates the oracle except that in this case, we do not have the gold standard. If we have N classifiers C_i, $i = 1$ to N, then we replace the gold standard by each of the N classifiers in succession, to get N outputs $Comb_i$, $i = 1$ to N.

The task of generating multiple classifier combination outputs that have a higher accuracy than the original classifiers is often considered to be easier than the task of determining the best of these outputs. We used the formulas in Equations 2, 3 and 4 to assign a score S_i to each of the N combination outputs $Comb_i$, and then we pick the one with the highest score. The function Sc (which corresponds to the function ϕ in Equation 1) gives the similarity between the entities in the pair (R, S).

$$S_i = \frac{1}{N-1} \sum_{\substack{j=1 \; to \; N \\ j \neq i}} Sc(Comb_i, C_j) \quad (2)$$

$$S_i = Sc(Comb_i, C_i) \quad (3)$$

$$S_i = \frac{1}{N-1} \sum_{\substack{j=1 \; to \; N \\ j \neq i}} Sc(Comb_i, Comb_j) \quad (4)$$

Entity-level sum-rule. We implemented a basic sum-rule at the entity level, where we generate only one combination classifier output by aligning the entities in the N classifiers and picking only one entity at each level of alignment. In the oracle, the reference for entity-alignment was the gold standard. Here, we use the baseline/full system (generated using the entire training and feature set) to do this. The entity-level alignment is represented as a table in Figure 2.

Let A_i, $i = 1$ to M be the aligned entities in one row of the table in Figure 2. Here, $M \leq N$ if

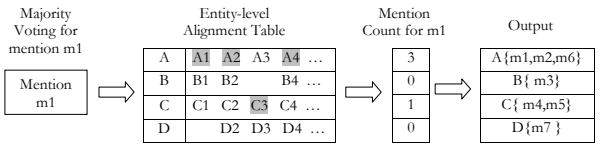

Figure 3: Mention-level majority voting

we exclude the baseline from the combination and $M \leq N + 1$ if we include it. To pick one entity out of these M entities, we use the traditional sum rule (Tulyakov *et al.*, 2008), shown in Equation 5, to compute the $S(A_i)$ for each A_i and pick the entity with the highest $S(A_i)$ value.

$$S(A_i) = \sum_{\substack{j=1 \; to \; N \\ j \neq i}} Sc(A_i, A_j) \quad (5)$$

2.4 Mention-level Majority Voting

In the previous techniques, entities are either picked or rejected as a whole but never broken down further. In the mention-level majority voting technique, we work at the mention level, so the entities created after combination may be different from the entities of all the classifiers that are being combined.

In the entity-level alignment table (shown in Figure 3), A, B, C and D refer to the entities in the baseline system and A1, A2, ..., D4 represent the entities of the input classifiers that are aligned with each of the baseline classifier entities. Majority voting is done by counting the number of times a mention is found in a set of aligned entities. So for every row in the table, we have a mention count. The row with the highest mention count is assigned the mention in the output. This is repeated for each mention in the document. In Figure 3, we are voting for the mention m1, which is found to have a voting count of 3 (the majority vote) at the entity-level A and a count of 1 at the entity-level C, so the mention is assigned to the entity A. It is important to note that some classifier entities may not align with any baseline classifier entity as we allow only a one-to-one mapping during alignment. Such entities will not be a part of the alignment table. If this number is large, it may have a considerable effect on the combination.

2.5 Document-level Boosting

Boosting techniques (Schapire, 1999) combine multiple classifiers, built iteratively and trained on re-weighted data, to improve classification accuracy. Since coreference resolution is done for a whole document, we can not split a document fur-

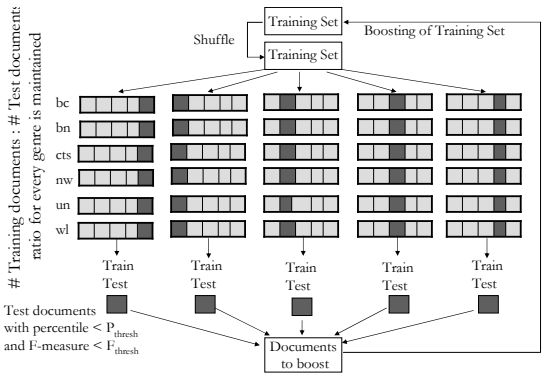

Figure 4: Document-level boosting

ther. So when we re-weight the training set, we are actually re-weighting the documents (hence the name document-level boosting). Figure 4 shows an overview of this technique.

The decision of which documents to boost is made using two thresholds: percentile threshold P_{thresh} and the F-measure threshold F_{thresh}. Documents in the test set that are in the lowest P_{thresh} percentile and that have a document F-measure less than F_{thresh} will be boosted in the training set for the next iteration. We shuffle the training set to create some randomness and then divide it into groups of training and test sets in a round-robin fashion such that a predetermined ratio of the number of training documents to the number of test documents is maintained. In Figure 4, the light gray regions refer to training documents and the dark gray regions refer to test documents. Another important consideration is that it is difficult to achieve good coreference resolution performance on documents of some genres compared to others, even if they are boosted significantly. In an iterative process, it is likely that documents of such genres will get repeatedly boosted. Also our training set has more documents of some genres and fewer of others. So we try to maintain, to some extent, the ratio of documents from different genres in the training set while splitting this training set further into groups of training and test sets.

3 Evaluation

This section describes the general setup used to conduct the experiments and presents an evaluation of the combination techniques that were implemented.

Experimental setup. The coreference resolution system used in our experiments makes use of a Maximum Entropy model which has lexical, syntactical, semantic and discourse features (Luo *et al.*, 2004). Experiments are conducted on ACE 2005 data (NIST, 2005), which consists of 599 documents from rich and diversified sources. We reserve the last 16% documents of each source as the test set, and use the rest of the documents as the training set. The ACE 2005 data split is tabulated in Table 1.

Table 1: Statistics of ACE 2005 data

DataSet	#Docs	#Words	#Mentions	#Entities
Training	499	253771	46646	16102
Test	100	45659	8178	2709
Total	599	299430	54824	18811

Table 2: Accuracy of generated and baseline classifiers

Classifier	Accuracy (%)
$C_1 - C_{15}$ Average	77.52
Highest	79.16
Lowest	75.81
C_0 Baseline	78.53

Bagging A total of 15 classifiers (C_1 to C_{15}) were generated, 12 of which were obtained by sampling the training set and the remaining 3 by sampling the feature set. We also make use of the baseline classifier C_0. The accuracy of C_0 to C_{15} has been summarized in Table 2. The agreement between the classifiers' output was found to be in the range of 93% to 95%. In this paper, the metric used to compute the accuracy of the coreference resolution is the Constrained Entity-Alignment F-Measure (CEAF) (Luo, 2005) with the entity-pair similarity measure in Equation 1.

Oracle. To conduct the oracle experiment, we train 1 to 15 classifiers and align their output to the gold standard. For all entities aligned with a gold entity, we pick the one with the highest score as the output. We measure the performance for varying number of classifiers, and the result is plotted in Figure 5.

First, we observe a steady and significant increase in CEAF for every additional classifier, because additional classifiers can only improve the alignment score. Second, we note that the oracle accuracy is 87.58% for a single input classifier C_1, *i.e.* an absolute gain of 9% compared to C_0. This is because the availability of gold entities makes it possible to remove many false-alarm entities. Finally, the oracle accuracy when all 15 classifiers are used as input is 94.59%, a 16.06% absolute improvement.

This experiment helps us to understand the performance bound of combining multiple classifiers and the contribution of every additional classifier.

Preliminary combination approaches. While the oracle results are encouraging, a natural question is

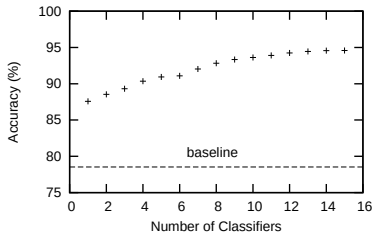

Figure 5: Oracle performance vs. number of classifiers

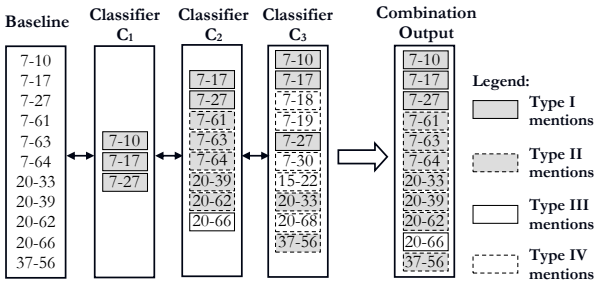

Figure 6: A real example showing the working of mention-level majority voting

how much performance gain can be attained if the gold standard is not available. To answer this question, we replace the gold standard with one of the classifiers C_1 to C_{15}, and align the classifiers. This is done in a round robin fashion as described in Section 2.3. The best performance of this procedure is 77.93%. The sum-rule combination output had an accuracy of 78.65% with a slightly different baseline of 78.81%. These techniques do not yield a statistically significant increase in CEAF but this is not surprising as C_1 to C_{15} are highly correlated.

Mention-level majority voting. This experiment is conducted to evaluate the mention-level majority voting technique. The results are not statistically better than the baseline, but they give us valuable insight into the working of the combination technique. The example in Figure 6 shows a single entity-alignment level for the baseline C_0 and 3 classifiers C_1, C_2, and C_3 and the combination output by mention-level majority voting. The mentions are denoted by the notation *'EntityID - MentionID'*, for example *7-10* is the mention with *EntityID=7* and *MentionID=10*. Here, we use the *EntityID* in the gold file. The mentions with *EntityID=7* are "correct" *i.e.* they belong in this entity, and the others are "wrong" *i.e.* they do not belong in this entity.

The aligned mentions are of four types:

- *Type I mentions* – These mentions have a highest voting count of 2 or more at the same entity-level alignment and hence appear in the output.
- *Type II mentions* – These mentions have a highest voting count of 1. But they are present in more than one input classifier and there is a tie between the mention counts at different entity-level alignments. The rule to break the tie is that mentions are included if they are also seen in the full system C_0. As can been seen, this rule brings in correct mentions such as *7-61, 7-63, 7-64*, but it also admits *20-33,20-39* and *20-62*. In the oracle, the gold standard helps to remove entities with false-alarm mentions, whereas the full system output is noisy and it is not strong enough to reliably remove undesired mentions.
- *Type III mentions* – There is only one mention *20-66* which is of this type. It is selected in the combination output since it is present in C_2 and the baseline C_0, although it has been rejected as a false-alarm in C_1 and C_3.
- *Type IV mentions* – These false-alarm mentions (relative to C_0) are rejected in the output. As can be seen, this correctly rejects mentions such as *15-22* and *20-68*, but it also rejects correct mentions *7-18, 7-19* and *7-30*.

In summary, the current implementation of this technique has a limited ability to distinguish correct mentions from wrong ones due to the noisy nature of C_0 which is used for alignment. We also observe that mentions spread across different alignments often have low-count and they are often tied in count. Therefore, it is important to set a minimum threshold for accepting these low-count majority votes and also investigate better tie-breaking techniques.

Document-level Boosting This experiment is conducted to evaluate the document-level boosting technique. Table 3 shows the results with the ratio of the number of training documents to the number of test documents equal to 80:20, F-measure threshold F_{thresh} = 74% and percentile threshold P_{thresh} = 25%. The accuracy increases by 0.7%, relative to the baseline. Due to computational complexity considerations, we used fixed values for the parameters. Therefore, these values may be suboptimal and may not correspond to the best possible increase in accuracy.

4 Related Work

A large body of literature related to statistical methods for coreference resolution is available (Ng and Cardie, 2003; Yang *et al.*, 2003; Ng, 2008; Poon and

Table 3: Results of document-level boosting

Iteration	Accuracy (%)
1	78.53
2	78.82
3	79.08
4	78.37

Domingos, 2008; McCallum and Wellner, 2003). Poon and Domingos (Poon and Domingos, 2008) use an unsupervised technique based on joint inference across mentions and Markov logic as a representation language for their system on both MUC and ACE data. Ng (Ng, 2008) proposed a generative model for unsupervised coreference resolution that views coreference as an EM clustering process. In this paper, we make use of a coreference engine similar to the one described by Luo *et al.* (Luo *et al.*, 2004), where a Bell tree representation and a Maximum entropy framework are used to provide a naturally incremental framework for coreference resolution. To the best of our knowledge, this is the first effort that utilizes classifier combination techniques to improve coreference resolution. Combination techniques have earlier been applied to various applications including machine translation (Jayaraman and Lavie, 2005), part-of-speech tagging (Brill and Wu, 1998) and base noun phrase identification (Sang *et al.*, 2000). However, the use of these techniques for coreference resolution presents a unique set of challenges, such as the issue of entity alignment between the multiple classifier outputs.

5 Conclusions and Future Work

In this paper, we examined and evaluated the applicability of bagging and boosting techniques to coreference resolution. We also provided empirical evidence that coreference resolution accuracy can potentially be improved by using multiple classifiers. In future, we plan to improve (1) the entity-alignment strategy, (2) the majority voting technique by setting a minimum threshold for the majority-vote and better tie-breaking, and (3) the boosting algorithm to automatically optimize the parameters that have been manually set in this paper. Another possible avenue for future work would be to test these combination techniques with other coreference resolution systems.

Acknowledgments

The authors would like to acknowledge Ganesh N. Ramaswamy for his guidance and support in conducting the research presented in this paper.

References

L. Breiman. 1996. Bagging predictors. In *Machine Learning*.

E. Brill and J. Wu. 1998. Classifier combination for improved lexical disambiguation. In *Proc. of COLING*.

J. Fiscus. 1997. A post-processing system to yield reduced word error rates: Recogniser output voting error reduction (rover). In *Proc. of ASRU*.

H. V. Halteren et al. 2001. Improving accuracy in word class tagging through the combination of machine learning systems. *Computational Linguistics*, 27.

S. Jayaraman and A. Lavie. 2005. Multi-engine machine translation guided by explicit word matching. In *Proc. of ACL*.

H. W. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2.

X. Luo et al. 2004. A mention-synchronous coreference resolution algorithm based on the bell tree. In *Proc. of ACL*.

X. Luo. 2005. On coreference resolution performance metrics. In *Proc. of EMNLP*.

A. McCallum and B. Wellner. 2003. Toward conditional models of identity uncertainty with application to proper noun coreference. In *Proc. of IJCAI/IIWeb*.

J. Munkres. 1957. Algorithms for the assignment and transportation problems. *Journal of the Society of Industrial and Applied Mathematics*, 5(1).

V. Ng and C. Cardie. 2003. Bootstrapping coreference classifiers with multiple machine learning algorithms. In *Proc. of EMNLP*.

V. Ng. 2008. Unsupervised models for coreference resolution. In *Proc. of EMNLP*.

NIST. 2005. ACE'05 evaluation. `www.nist.gov/speech/tests/ace/ace05/index.html`.

H. Poon and P. Domingos. 2008. Joint unsupervised coreference resolution with Markov Logic. In *Proc. of EMNLP*.

E. F. T. K. Sang et al. 2000. Applying system combination to base noun phrase identification. In *Proc. of COLING 2000*.

R.E. Schapire. 1999. A brief introduction to boosting. In *Proc. of IJCAI*.

S. Tulyakov et al. 2008. Review of classifier combination methods. In *Machine Learning in Document Analysis and Recognition*.

X. Yang et al. 2003. Coreference resolution using competition learning approach. In *Proc. of ACL*.

Solving the "Who's Mark Johnson" Puzzle: Information Extraction Based Cross Document Coreference

Jian Huang[†*] **Sarah M. Taylor**[‡] **Jonathan L. Smith**[‡] **Konstantinos A. Fotiadis**[§] **C. Lee Giles**[†]
[†]Information Sciences and Technology
Pennsylvania State University, University Park, PA 16802, USA
[‡§]Advanced Technology Office, Lockheed Martin IS&GS
[‡]4350 N. Fairfax Drive, Suite 470, Arlington, VA 22203, USA
[§]230 Mall Blvd, King of Prussia, PA 19406, USA

Abstract

Cross Document Coreference (CDC) is the problem of resolving the underlying identity of entities across multiple documents and is a major step for document understanding.

We develop a framework to efficiently determine the identity of a person based on extracted information, which includes unary properties such as gender and title, as well as binary relationships with other named entities such as co-occurrence and geo-locations. At the heart of our approach is a suite of similarity functions (specialists) for matching relationships and a relational density-based clustering algorithm that delineates name clusters based on pairwise similarity. We demonstrate the effectiveness of our methods on the WePS benchmark datasets and point out future research directions.

1 Introduction

The explosive growth of web data offers users both the opportunity and the challenge to discover and integrate information from disparate sources. As alluded to in the title, a search query of the common name "Mark Johnson" refers to as many as 70 namesakes in the top 100 search results from the Yahoo! search engine, only one of whom is the Brown University professor and co-author of an ACL 2006 paper (see experiments). Cross document coreference (CDC) (Bagga and Baldwin, 1998) is a distinct technology that consolidates named entities *across* documents according to their real referents. Despite the variety of styles and content in different text, CDC can break the boundaries of documents and cluster those mentions referring to the same Mark Johnson. As unambiguous person references are key to many tasks, e.g. social network analysis, this work focuses on person named entities. The method can be later extended to organizations.

We highlight the key differences between our proposed CDC system with past person name search systems. First, we seek to transcend the simple bag of words approaches in earlier CDC work by leveraging state-of-the-art information extraction (IE) tools for disambiguation. The main advantage is that our IE based approach has access to accurate information such as a person's work titles, geo-locations, relationships and other attributes. Traditional IR approaches, on the other hand, may naively use the terms in a document which can significantly hamper accuracy. For instance, an article about Hillary Clinton may contain references to journalists, politicians who make comments about her. Even with careful word selection, such textual features may still confuse the disambiguation system about the true identity of the person. The information extraction process in our work can thus be regarded as an intelligent feature selection step for disambiguation. Second, after coreferencing, our system not only yields clusters of documents, but also structured information which is highly useful for automated document understanding and data mining.

We review related work on CDC next and describe our approach in Section 3. The methods are evaluated on benchmark datasets in Section 4. We discuss directions for future improvement in Section 5 and conclude in Section 6.

2 Related Work

There is a long tradition of work on the within document coreference (WDC) problem in NLP,

*Contact author: jhuang@ist.psu.edu

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 7–12,
Boulder, Colorado, June 2009.

which links named entities with the same referent *within* a document into a WDC chain. State-of-the-art WDC systems, e.g. (Ng and Cardie, 2001), leverage rich lexical features and use supervised and unsupervised machine learning methods.

Research on cross document coreference began more recently. (Bagga and Baldwin, 1998) proposed a CDC system to merge the WDC chains using the Vector Space Model on the summary sentences. (Gooi and Allan, 2004) simplified this approach by eliminating the WDC module without significant deterioration in performance. Clustering approaches (e.g. hierarchical agglomerative clustering (Mann and Yarowsky, 2003)) have been commonly used for CDC due to the variety of data distributions of different names. Our work goes beyond the simple co-occurrence features (Bagga and Baldwin, 1998) and the limited extracted information (e.g. biographical information in (Mann and Yarowsky, 2003) that is relatively scarce in web data) using the broad range of relational information with the support of information extraction tools. There are also other related research problems. (Li et al., 2004) solved the robust reading problem by adopting a probabilistic view on how documents are generated and how names are sprinkled into them. Our previous work (Huang et al., 2006) resolved the author name ambiguity problem based on the metadata records extracted from academic papers.

3 Methods

The overall framework of our CDC system works as follows. Given a document, the information extraction tool first extracts named entities and constructs WDC chains. It also creates linkages (relationships) between entities. The similarity between a pair of relationships in WDC chains is measured by an *awakened* similarity specialist and the similarity between two WDC chains is determined by the mixture of awakened specialists' predictions. Finally, a density-based clustering method generates clusters corresponding to real world entities. We describe these steps in detail.

3.1 Entity and Relationship Extraction

Given a document, an information extraction tool is first used to extract named entities and perform within document coreference. Hence, named entities in each document are divided into a set of WDC chains, each chain corresponding to one real world entity. In addition, state-of-the-art IE tools are capable of creating relational information between named entities. We use an IE tool AeroText[1] (Taylor, 2004) for this purpose. Besides the attribute information about the person named entity (first/middle/last names, gender, mention, etc), AeroText also extracts relationship information between named entities, such as Family, List, Employment, Ownership, Citizen-Resident-Religion-Ethnicity, etc, as specified in the Automatic Content Extraction (ACE) evaluation. The input to the CDC system is a set of WDC chains (with relationship information stored in them) and the CDC task is to merge these WDC chains[2].

3.2 Similarity Features

We design a suite of similarity functions to determine whether the relationships in a pair of WDC chains match, divided into three groups:

Text similarity. To decide whether two names in the co-occurrence or family relationship match, we use SoftTFIDF (Cohen et al., 2003), which has shown best performance among various similarity schemes tested for name matching. SoftTFIDF is a hybrid matching scheme that combines the token-based TFIDF with the Jaro-Winkler string distance metric. This permits inexact matching of named entities due to name variations, spelling errors, etc.

Semantic similarity. Text or syntactic similarity is not always sufficient for matching relationships. For instance, although the mentions "U.S. President" and "Commander-in-chief" have no textual overlap, they are semantically highly related as they can be synonyms. We use WordNet and the information theory based JC semantic distance (Jiang and Conrath, 1997) to measure the semantic similarity between concepts in relationships such as mention, employment, ownership and so on.

[1] AeroText is a text mining application for content analysis, with main focus on information extraction including entity extraction and intrasource link analysis (see `http://en.wikipedia.org/wiki/AeroText`).

[2] We make no distinctions whether WDC chains are extracted from the same document. Indeed, the CDC system can correct the WDC errors due to lack of information for merging named entities within a document.

Other rule-based similarity. Several other cases require special treatment. For example, the employment relationships of *Senator* and *D-N.Y.* should match based on domain knowledge. Also, we design rule-based similarity functions to handle nicknames (Bill and William), acronyms (COLING for International Conference on Computational Linguistics), and geographical locations[3].

3.3 Learning a Similarity Matrix

After the similarity features between a pair of WDC chains are computed, we need to compute the pairwise distance metric for clustering. (Cohen et al., 2003) trained a binary SVM model and interpreted its confidence in predicting the negative class as the distance metric. In our case of using information extraction results for disambiguation, however, only some of the similarity features are present based on the availability of relationships in two WDC chains. Therefore, we treat each similarity function as a subordinate predicting algorithm (called specialist) and utilize the specialist learning framework (Freund et al., 1997) to combine the predictions. Here, a specialist is *awake* only when the same relationships are present in two WDC chains. Also, a specialist can refrain from making a prediction for an instance if it is not confident enough. In addition to the similarity scores, specialists have different weights, e.g. a match in a family relationship is considered more important than in a co-occurrence relationship.

The Specialist Exponentiated Gradient (SEG) (Freund et al., 1997) algorithm is adopted to learn to mix the specialists' prediction. Given a set of T training instances $\{\mathbf{x_t}\}$ ($x_{t,i}$ denotes the i-th specialist's prediction), the SEG algorithm minimizes the square loss of the outcome $\tilde{y}$ in an online manner (Algorithm 1). In each learning iteration, SEG first predict $\tilde{y}_t$ using the set of awake experts E_t with respect to instance $\mathbf{x}_t$. The true outcome y_t (1 for coreference and 0 otherwise) is then revealed and square loss L is incurred. SEG then updates the weight distribution $\mathbf{p}$ accordingly.

To sum up, the similarity between a pair of WDC chains w_i and w_j can be represented in a similarity matrix $\mathcal{R}$, with $r_{i,j}$ computed by the SEG prediction step using the learned weight distribution $\mathbf{p}$ (Equation 1). A relational clustering algorithm then clusters entities using $\mathcal{R}$, as we introduce next.

Algorithm 1 SEG (Freund et al., 1997)

Input: Initial weight distribution $\mathbf{p^1}$;
learning rate $\eta > 0$; training set $\{\mathbf{x_t}\}$

1: **for** t=1 to T **do**
2: Predict using:

$$\tilde{y}_t = \frac{\sum_{i \in E_t} p_i^t x_{t,i}}{\sum_{i \in E_t} p_i^t} \quad (1)$$

3: Observe true label y_t and incur square loss $L(\tilde{y}_t, y_t) = (\tilde{y}_t - y_t)^2$
4: Update weight distribution: for $i \in E_t$

$$p_i^{t+1} = p_i^t e^{-2\eta x_{t,i}(\tilde{y}_t - y_t)} \frac{\sum_{j \in E_t} p_j^t}{\sum_{j \in E_t} p_j^t e^{-2\eta x_{t,i}(\tilde{y}_t - y_t)}}$$

$p_i^{t+1} = p_i^t$, otherwise
5: **end for**

Output: Model $\mathbf{p}$

3.4 Relational Clustering

The set of WDC chains to be clustered are represented by a relational similarity matrix. Most of the work in clustering, however, is only capable of clustering numerical object data (e.g. K-means). Relational clustering algorithms, on the other hand, cluster objects based on the less direct measurement of similarity between object pairs. We choose to use a density based clustering algorithm DBSCAN (Ester et al., 1996) mainly for two reasons.

First, most clustering algorithm require the number of clusters K as an input parameter. The optimal K can apparently vary greatly for names with different frequency and thus is a sensitive parameter. Even if a cluster validity index is used to determine K, it usually requires running the underlying clustering algorithm multiple times and hence is inefficient for large scale data. DBSCAN, as a density based clustering method, only requires density parameters such as the radius of the neighborhood ϵ that are universal for different datasets. As we show in the experiment,

[3]Though a rich set of similarity features has been built for matching the relationships, they may not encompass all possible cases in real world documents. The goal of this work, however, is to focus on the algorithms instead of knowledge engineering.

density parameters are relatively insensitive for disambiguation performance.

Second, the distance metric in relational space may be non-Euclidean, rendering many clustering algorithms ineffective (e.g. single linkage clustering algorithm is known to generate chain-shaped clusters). Density-based clustering, on the other hand, can generate clusters of arbitrary shapes since only objects in dense areas are placed in a cluster.

DBSCAN induces a density-based cluster by the core objects, i.e. objects having more than a specified number of other data objects in their neighborhood of size ϵ. In each clustering step, a seed object is checked to determine whether it's a core object and if so, it induces other points of the same cluster using breadth first search (otherwise it's considered as a noise point). In interest of space, we refer readers to (Ester et al., 1996) for algorithmic details of DBSCAN and now turn our attention to evaluating the disambiguation performance of our methods.

4 Experiments

We first formally define the evaluation metrics, followed by the introduction to the benchmark test sets and the system's performance.

4.1 Evaluation Measures

We evaluate the performance of our method using the standard purity and inverse purity clustering metrics. Let a set of clusters $\mathcal{C} = \{C_1, ..., C_s\}$ denote the system's output and a set of categories $\mathcal{D} = \{D_1, ..., D_t\}$ be the gold standard. Both $\mathcal{C}$ and $\mathcal{D}$ are partitions of the WDC chains $\{w_1, ..., w_n\}$ ($n = \sum_i |C_i| = \sum_j |D_j|$). First, the precision of a cluster C_i *w.r.t.* a category D_j is defined as,

$$\text{Precision}(C_i, D_j) = \frac{|C_i \cap D_j|}{|C_i|}$$

Purity is defined as the weighted average of the maximum precision achieved by the clusters on one of the categories,

$$\text{Purity}(\mathcal{C}, \mathcal{D}) = \sum_{i=1}^{s} \frac{|C_i|}{n} \max_j \text{Precision}(C_i, D_j)$$

Hence purity penalizes putting noise WDC chains in a cluster. Trivially, the maximum purity (i.e. 1) can be achieved by making one cluster per WDC chain (referred to as the one-in-one baseline).

Reversing the role of clusters and categories, $\text{Inverse\_purity}(\mathcal{C}, \mathcal{D}) \stackrel{def}{=} \text{Purity}(\mathcal{D}, \mathcal{C})$. Inverse Purity penalizes splitting WDC chains belonging to the same category into different clusters. The maximum inverse purity can be achieved by putting all chain in one cluster (all-in-one baseline).

Purity and inverse purity are similar to the precision and recall measures commonly used in information retrieval. There is a tradeoff relationship between the two and their harmonic mean $F_{0.5}$ is used for performance evaluation.

4.2 Datasets

We evaluate our methods using the benchmark test collection from the ACL SemEval-2007 web person search task (WePS hereafter) (Artiles et al., 2007). The test collection consists of three sets of documents for 10 different names, sampled from the English Wikipedia (famous people), participants of the ACL 2006 conference (computer scientists) and common names from the US Census data, respectively. For each ambiguous name, the top 100 documents retrieved from the Yahoo! Search API were annotated by human annotators according to the actual entity of the name. This yields on average 45 different real world entities per set and about 3k documents in total.

We note that the annotation in WePS makes the simplifying assumption that each document refers to only one real world person among the namesakes in question. The CDC task in the perspective of this paper, however, is to merge the WDC chains rather than documents. Hence in our evaluation, we adopt the document label to annotate the WDC chain from the document that corresponds to the person name search query. Despite the difference, the results of the one-in-one and all-in-one baselines are almost identical to those reported in the WePS evaluation ($F_{0.5} = 0.61, 0.40$ respectively). Hence the performance reported here is comparable to the official evaluation results (Artiles et al., 2007).

4.3 Experiment Results

We computed the similarity features from the WDC chains extracted from the WePS training data and subsampled the non-coreferent pairs to generate a

Table 1: Cross document coreference performance (macro-averaged scores, I-Pur denotes inverse purity).

Test set	Method	Purity	I-Pur	$F_{0.5}$
Wikipedia	AT-CDC	0.684	0.725	0.687
ACL-06	AT-CDC	0.792	0.657	0.712
US Census	AT-CDC	0.772	0.700	0.722
Global Average	AT-CDC	0.749	0.695	0.708
	One-in-one	1.000	0.482	0.618
	All-in-one	0.279	1.000	0.389

training set of around 32k pairwise instances. We then used the SEG algorithm to learn the weight distribution model. The macro-averaged cross document coreference results on the WePS test sets are reported in Table 1. The $F_{0.5}$ score of our CDC system (AT-CDC) is 0.708, comparable to the test results of the first tier systems in the official evaluation. The two baselines are also included. Because the test set is very ambiguous (on average only two documents per real world entity), the one-in-one baseline has relatively high $F_{0.5}$ score.

The Wikipedia, ACL06 and US Census sets have on average 56, 31 and 50 entities per name respectively. We notice that as the data set becomes more ambiguous, purity decreases implying it's harder for the system to discard irrelevant documents from a cluster. The other case is true for inverse purity. In particular, we are interested in how the coreference performance changes with the number of entities per name (which can be viewed as the ambiguity level of a data set). This is shown in Figure 1. We observe that in general the harmonic mean of the purity is fairly stable across different number of entities per dataset (generally within the band between 0.6 and 0.8). This is important because the system's performance does not vary greatly with the underlying data characteristics. There is a particular name (with only one underlying referent) that appears to be an outlier in performance in Figure 1. After examining the extraction results, we notice that the extracted relationships refer to the same person's employment, coauthors and geo-locations. The generated CDC clusters correctly reflect the different aspects of the person but the system is unable to link them together due to the lack of information for merging. This motivates us to further improve performance in future work.

Figure 2 shows how the coreference performance

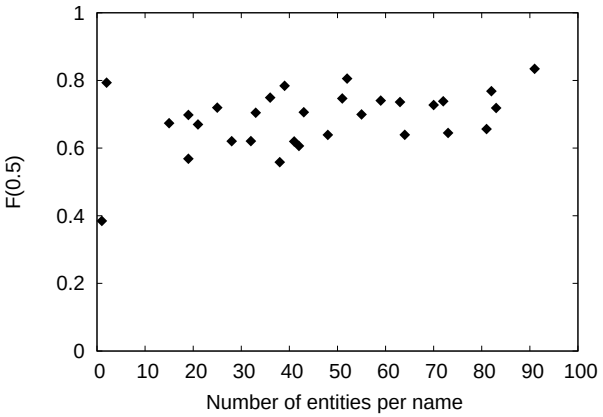

Figure 1: Coreference performance for names with different number of real world entities.

changes with different density parameter ϵ. We observe that as we increase the size of the ϵ neighborhood, inverse purity increases indicating that more correct coreference decisions are made. On the other hand, purity decreases as more noise WDC chains appear in clusters. Due to this tradeoff relationship, the F score is fairly stable with a wide range of ϵ values and hence the density parameter is rather insensitive (compared to, say, the number of clusters K).

5 Future Work

We see several opportunities to improve the coreference performance of the proposed methods.

First, though the system's performance compares favorably with the WePS submissions, we observe that purity is higher than inverse purity, indicating that the system finds it more difficult to link coreferent documents than to discard noise from

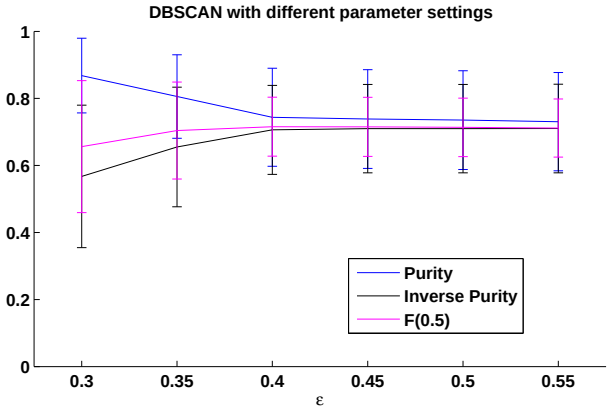

Figure 2: Coreference performance with different ϵ.

clusters. Thus coreferencing based solely on the information generated by an information extraction tool may not always be sufficient. For one, it remains a huge challenge to develop a general purpose information extraction tool capable of applying to web documents with widely different formats, styles, content, etc. Also, even if the extraction results are perfect, relationships extracted from different documents may be of different types (family memberships vs. geo-locations) and cannot be directly matched against one another. We are exploring several methods to complement the extracted relationships using other information:

• *Context-aided CDC*. The context where an named entity is extracted can be leveraged for coreference. The bag of words in the context tend to be less noisy than that from the entire document. Moreover, we can use noun phrase chunkers to extract base noun phrases from the context. These word or phrase level features can serve as a safenet when the IE tool fails.

• *Topic-based CDC*. Similar to (Li et al., 2004), document topics can be used to ameliorate the sparsity problem. For example, the topics *Sport* and *Education* are important cues for differentiating mentions of "Michael Jordan", which may refer to a basketball player, a computer science professor, etc.

Second, as noted in the top WePS run (Chen and Martin, 2007), feature development is important in achieving good coreference performance. We aim to improve the set of similarity specialists in our system by leveraging large knowledge bases.

Moreover, although the CDC system is developed in the web person search context, the methods are also applicable to other scenarios. For instance, there is tremendous interest in building structured databases from unstructured text such as enterprise documents and news articles for data mining, where CDC is a key step for "understanding" documents from disparate sources. We plan to continue our investigations along these lines.

6 Conclusions

We have presented and implemented an information extraction based Cross Document Coreference (CDC) system that employs supervised and unsupervised learning methods. We evaluated the proposed methods with experiments on a large benchmark disambiguation collection, which demonstrate that the proposed methods compare favorably with the top runs in the SemEval evaluation. We believe that by incorporating information such as context and topic, besides the extracted relationships, the performance of the CDC can be further improved. We have outlined research plans to address this and several other issues.

References

Javier Artiles, Julio Gonzalo, and Satoshi Sekine. 2007. The SemEval-2007 WePS evaluation: Establishing a benchmark for the web people search task. In *Proc 4th Int'l Workshop on Semantic Evaluations (SemEval)*.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proc. of 36th ACL and 17th COLING*.

Ying Chen and James Martin. 2007. Towards robust unsupervised personal name disambiguation. In *Proceedings of EMNLP and CoNLL*, pages 190–198.

William W. Cohen, Pradeep Ravikumar, and Stephen E. Fienberg. 2003. A comparison of string distance metrics for name-matching tasks. In *Proc. of IJCAI Workshop on Information Integration on the Web*.

Martin Ester, Hans-Peter Kriegel, Jorg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of 2nd KDD*, pages 226 – 231.

Yoav Freund, Robert E. Schapire, Yoram Singer, and Manfred K. Warmuth. 1997. Using and combining predictors that specialize. In *Proceedings of 29th ACM symposium on Theory of computing (STOC)*.

Chung H. Gooi and James Allan. 2004. Cross-document coreference on a large scale corpus. In *HLT-NAACL*.

Jian Huang, Seyda Ertekin, and C. Lee Giles. 2006. Efficient name disambiguation for large scale databases. In *Proc. of 10th PKDD*, pages 536 – 544.

Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of ROCLING X*.

Xin Li, Paul Morie, and Dan Roth. 2004. Robust reading: Identification and tracing of ambiguous names. In *Proceedings of HLT-NAACL*, pages 17–24.

Gideon S. Mann and David Yarowsky. 2003. Unsupervised personal name disambiguation. In *Proceedings of HLT-NAACL*, pages 33–40.

Vincent Ng and Claire Cardie. 2001. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th ACL*, pages 104–111.

Sarah M. Taylor. 2004. Information extraction tools: Deciphering human language. *IT Professional*, 6(6):28 – 34.

Exploring Topic Continuation Follow-up Questions using Machine Learning

Manuel Kirschner
KRDB Center
Faculty of Computer Science
Free University of Bozen-Bolzano, Italy
kirschner@inf.unibz.it

Raffaella Bernardi
KRDB Center
Faculty of Computer Science
Free University of Bozen-Bolzano, Italy
bernardi@inf.unibz.it

Abstract

Some of the Follow-Up Questions (FU Q) that an Interactive Question Answering (IQA) system receives are not topic shifts, but rather continuations of the previous topic. In this paper, we propose an empirical framework to explore such questions, with two related goals in mind: (1) modeling the different relations that hold between the FU Q's answer and either the FU Q or the preceding dialogue, and (2) showing how this model can be used to identify the correct answer among several answer candidates. For both cases, we use Logistic Regression Models that we learn from real IQA data collected through a live system. We show that by adding dialogue context features and features based on sequences of domain-specific actions that represent the questions and answers, we obtain important additional predictors for the model, and improve the accuracy with which our system finds correct answers.

1 Introduction

Interactive Question Answering (IQA) can be described as a fusion of the QA paradigm with dialogue system capabilities. While classical QA is concerned with questions posed in isolation, its interactive variant is intended to support the user in finding the correct answer via natural-language dialogue. In an IQA setting, both the system and the user can pose Follow-Up Questions (FU Q). In the second case, whenever an IQA system receives an additional user question (note that this is what we call a *Follow-Up Question* throughout this work), it can either interpret it as being thematically related to a previous dialogue segment (*topic continuation*), or as a shift to some new, unrelated topic (*topic shift*). A definition of thematic relatedness of FU Qs might rely on the elements of the attentional state, i.e., on the objects, properties and relations that are salient before and after processing the user question. Topic continuation FU Qs should be interpreted within the context, whereas topic shift FU Qs have to be treated as first questions and can thus be processed with standard QA technologies. Therefore, a **first task** in IQA is to detect whether a FU Q is a topic shift or a topic continuation (Yang et al., 2006).

To help answering topic continuation FU Qs, an IQA system would need to fuse the FU Q with certain information from the dialogue context (cf. (van Schooten et al., 2009)). Thus, a **second task** in IQA is to understand which turns in the dialogue context are possible locations of such information, and exactly what kind of information should be considered. Knowing that a FU Q concerns the same topic as the previous question or answer, we thus want to study in more detail the way the informational content of questions and answers evolves before/after the FU Q is asked. A model of these so-called *informational transitions* would provide insights into what a user is likely to ask about next in natural coherent human-machine dialogue.

In order to tackle any of the two IQA tasks mentioned above we need IQA dialogues. Most current work on IQA uses the TREC QA data; the TREC QA tracks in 2001 and 2004 included series of context questions, where FU Qs always depended on the context set by an earlier question from the same series. However, these data were constructed artificially and are not representative of actual dialogues from an IQA system (for instance, system answers are not considered at all). Real IQA data yield chal-

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 13–18,
Boulder, Colorado, June 2009.

lenges for an automatic processing approach (Yang et al., 2006). Our work is based on collecting and analyzing IQA dialogues from users of a deployed system.

In this paper, we address the second task introduced above, namely the study of common relations between the answer to a topic continuation FU Q and other turns in the dialogue context. Our collected dialogue data are from the "library help desk" domain. In many of the dialogues, library users request information about a specific library-related action; we are thus dealing with task-oriented dialogues. This work is based on two hypotheses regarding relations holding between the FU Q's answer and the dialogue context. For studying such relations, we want to explore the usefulness of (1) a representation of the library-related action underlying questions and answers, and (2) a representation of the dialogue context of the FU Q.

2 Background

In order to understand what part of the history of the dialogue is important for processing FU Qs, significant results come from Wizard-of-Oz studies, like (Dahlbäck and Jönsson, 1989; Bertomeu et al., 2006; Kirschner and Bernardi, 2007), from which it seems that the immediate linguistic context (i.e., the last user initiative plus the last system response) provides the most information for resolving any context-dependency of the FU Qs. These studies analyzed one particular case of topic continuation FU Q, namely those questions containing reference-related discourse phenomena (ellipsis, definite description or anaphoric pronoun); we assume that the results could be extended to fully specified questions, too.

Insights about the informational transitions within a dialogue come from Natural Language Generation research. (McCoy and Cheng, 1991) provide a list of informational transitions (they call them focus shifts) that we can interpret as transitions based on certain thematic relations. Depending on the conversation's current focus type, they list specific focus shift candidates, i.e., the items that should get focus as a coherent conversation moves along. Since we are interested in methods for interpreting FU Qs automatically, we decided to restrict ourselves to use only the "action" focus type to represent the focus of questions and answers in IQA dialogues. We conjecture that actions form a suitable and robust basis for describing the (informational) meaning of utterances in our class of task-based "help desk" IQA dialogues. Table 1 shows the focus shift candidates for a current focus of type "action". In this work we concentrate on the informational transitions involving *two actions* (i.e., including one of the focus targets listed in bold face in the table).

Node type	Informational transition targets
Action	Actor, object, etc., of the action – any participant (Fillmore) role; purpose (goal) of action, **next action in some sequence, subactions, specializations of the action**

Table 1: Possible informational transition targets for "action" node type (McCoy and Cheng, 1991)

3 Exploring topic continuation FU Qs using Machine Learning

We base our study of topic continuation FU Qs on the two main results described in Section 2: We study snippets of dialogues consisting of four turns, viz. a user question (Q_{-1}), the corresponding system answer (A_{-1}), the FU Q and its system answer (A_0); we use Logistic Regression Models to learn from these snippets (1) which informational (action-action) transitions hold between A_0 and the FU Q or the preceding dialogue, and (2) how to predict whether a specific answer candidate A_0 is correct for a given dialogue snippet.

3.1 Machine learning framework: Logistic Regression

Logistic regression models (Agresti, 2002) are generalized linear models that describe the relationship between features (predictors) and a binary outcome (in our case: answer correctness). We estimate the model parameters (the beta coefficients $\beta_1, \ldots, \beta_k$) that represent the contribution of each feature to the total answer correctness score using maximum likelihood estimation. Note that there is a close relationship to Maximum Entropy models, which have performed well in many tasks. A major advantage of using logistic regression as a supervised machine

learning framework (as opposed to other, possibly better performing approaches) is that the learned coefficients are easy to interpret. The logistic regression equation which predicts the probability for a particular answer candidate A_0 being correct, depending on the learned intercept β_0, the other beta coefficients and the feature values $x_1, \ldots, x_k$ (which themselves depend on a combination of Q_{-1}, A_{-1}, FU Q or A_0) is:

$$\text{Prob}\{\text{answerCorrect}\} = \frac{1}{1+\exp(-X\hat{\beta})}, \text{ where}$$

$$X\hat{\beta} \quad = \quad \beta_0 + (\beta_1 x_1 + \ldots + \beta_k x_k)$$

3.2 Dialogue data collection

We have been collecting English human-computer dialogues using BoB, an IQA system which is publicly accessible on the Library's web-site of our university[1]. We see the availability of dialogue data from genuinely motivated visitors of the library web-site as an interesting detail of our approach; our data are less constrained and potentially more difficult to interpret than synthesized dialogues (e.g., TREC context track data), but should on the other hand provide insights into the structure of actual IQA dialogues that IQA systems might encounter. We designed BoB as a simple chatbot-inspired application that robustly matches user questions using regular expression-based question patterns, and returns an associated canned-text answer from a repository of 529. The question patterns and answers have been developed by a team of librarians, and cover a wide range of library information topics, e.g., opening time, lending procedures and different library services. In the context of this work, we use BoB merely as a device for collecting real human-computer IQA dialogues.

As a preliminary step towards automatically modeling action-based informational transitions triggered by FU Qs, we annotated each of the 529 answers in our IQA system's repository with the "library action" that we considered to best represent its (informational) meaning. For this, we had devised a (flat) list of 25 library-related actions by analyzing the answer repository (e.g.: access, borrow, change, deliver). We also added synonymous verbs to our action list, like "obtain" for "borrow". If we did not find any action to represent a system answer, we assigned it a special "generic-information" tag, e.g. for answers to questions like "What are the opening times?".

We base our current study on the dialogues collected during the first four months of the IQA system being accessible via the Library's web site. After a first pass of manually filtering out dialogues that consisted only of a single question, or where the question topics were only non-library-related, the collected corpus consists of 948 user questions (first or FU Qs) in 262 dialogue sessions (i.e., from different web sessions). We hand-annotated the user FU Qs in these dialogues as either "topic continuation" (248 questions), or "topic shift" (150 questions).

The remaining FU Qs are user replies to system-initiative clarification questions, which we do not consider here. For each user question, we marked whether the answer given by the IQA system was correct; in the case of wrong answers, we asked our library domain experts to provide the correct answer that BoB should have returned. However, we only corrected the system answer in those cases where the user did not ask a further FU Q afterwards, as we must not change on-going dialogues.

To get the actual training/test data, we had to further constrain the set of 248 topic continuation FU Qs. We removed all FU Qs that immediately follow a system answer that we considered incorrect; this is because any further FU Q is then uttered in a situation where the user is trying to react to the problematic answer, which clearly influences the topic of the FU Q. Of the then remaining 76 FU Qs, we keep the following representation of the dialogue context: the previous user question Q_{-1} and the previous system answer A_{-1}. We also keep the FU Q itself, and its corresponding correct answer A_0.

Finally, we automatically annotated each question with one or more action tags. This was done by simply searching the stemmed question string for any verb stem from our list of 25 actions (or one of their synonyms); if no action stem is found, we assigned the "generic-information" tag to the question. Note that this simple action detection algorithm for questions fails in case of context-dependent questions where the verb is elided or if the question contains still unknown action synonyms.

[1] www.unibz.it/library

3.3 Features

In the machine learning framework introduced above, the model is intended to predict the correctness of a given system answer candidate, harnessing information from the local dialogue context: Q_{-1}, A_{-1}, FU Q and the particular answer candidate A_0. We now introduce different features that relate A_0 to either the FU Q or some other preceding turn of the dialogue. The features describe specific aspects of how the answer candidate relates to the current dialogue. Note that we do not list features relating Q_{-1} and A_0, since our experiments showed no evidence for including them in our models.

tfIdfSimilarityQA, tfIdfSimilarityAA: TF/IDF-based proximity scores (ranging from 0 to 1) between two strings, namely FU Q and A_0, or A_{-1} and A_0, respectively. Based on vector similarity (using the cosine measure of angular similarity) over dampened and discriminatively weighted term frequencies. Definition of the TF/IDF distance: two strings are more similar if they contain many of the same tokens with the same relative number of occurrences of each. Tokens are weighted more heavily if they occur in few documents[2], hence we used a subset of the UK English version of the Web-as-Corpus data[3] to train the IDF scores.

Features based on action sequences. To describe the action-related informational transitions we observe between the FU Q and A_0 and between A_{-1} and A_0, we use two sets of features, both of which are based on *hand-annotated* actions for answers and *automatically assigned* actions for questions. **actionContinuityQA, actionContinuityAA:** simple binary features indicating whether *the same* library action (or one of its synonyms) was identified between the FU Q and A_0, or A_{-1} and A_0, respectively. **lmProbQA, lmProbAA:** encode Statistical Language Model probabilities for action tag sequences, i.e., the probability for A_0 having a certain action, given the action associated with FU Q, or the action of A_{-1}, respectively. The underlying Statistical Language Models are probability distributions over action-action sequences that reflect how likely certain action sequences occur in our IQA dialogues, thus capturing properties of salient action sequences. More technically, we use Witten-Bell smoothed 2-gram statistical language models, which we trained on our action-tagged FU Q data.

[2]Cf. Alias-i's LingPipe documentation `http://alias-i.com/lingpipe/demos/tutorial/stringCompare/read-me.html`

[3]`http://wacky.sslmit.unibo.it`

4 Results

For the evaluation of the logistic regression model, we proceed as follows. Applying a cross-validation scheme, we split our 76 FU Q training examples randomly into five non-intersecting partitions of 15 (or 16) FU Q (with corresponding Q_{-1}, A_{-1}, and correct A_0) each. To train the logistic regression model, we need training data consisting of a vector of independent variables (the various feature values), along with the binary dependent variable, i.e., "answer correct" or "answer false". We generate these training data by "multiplying out" each training partition's 61 FU Qs (76 minus the held-out test set of 15) with all 529 answer candidates; for each FU Q dialogue snippet used for training, this results in one positive training example (where A_0 is the 1 correct out 529 answer candidates), and 528 negative training examples (for all other answer candidates).

For each of the five training/test partitions, we train a different model. We then evaluate each of these models on their corresponding held-out test set. Following the cross-validation idea through, we also train separate Statistical Language Models on sequences of action tags for each of the five training splits; this ensures that the language model probabilities were never trained on test data. We perform the evaluation in terms of the mean rank that the correct answer A_0 is assigned after ranking all 529 answer candidates (by evaluating the logistic regression equation to yield answer scores).

In the following, we give details of different logistic regression models we experimented with. Initially, we chose a subset from the list of features introduced above. Our goal was to retain as few features as needed to explore our two hypotheses, i.e., whether we can make use of (1) a representation of the FU Q's underlying library action, and/or (2) a representation of the immediate dialogue context. By dropping uninformative features, the result-

ing models become simpler and easier to interpret. With this goal in mind, we applied a fast backwards elimination routine that drops uninformative predictors (cf. (Baayen, 2008, p.204)) on the five training data splits. In all five splits, both TF/IDF features turned out to be important predictors; in four of the splits, also lmProbQA was retained. lmProbAA was dropped as superfluous in all but two splits, and actionSimilarityAA was retained only in one. With these results, the set of features we retain for our modeling experiments is: tfIdfSimilarityQA, tfIdfSimilarityAA and lmProbQA.

"Complete" model: tfIdfSimilarityQA, tfIdfSimilarityAA and lmProbQA We estimated logistic regression models on the five cross evaluation training sets using all three features as predictors. Table 2 shows the mean ranks of the correct answer for the five evaluation runs, and an overall mean rank with the average across the five splits.

To illustrate the contribution of each of the three predictors towards the score of an answer candidate, we provide the (relevant linear part of) the learned logistic regression equation for the "complete" model (trained on split 1 of the data). Note that the "answer ranker" evaluates this equation to get a score for an answer candidate A_0.

$$\begin{aligned} X\hat{\beta} &= -8.4 + (9.5 * \text{tfIdfSimilarityQA} + \\ &\quad 4.6 * \text{tfIdfSimilarityAA} + \\ &\quad 1.7 * \text{lmProbQA}) \end{aligned}$$

Reduced model 1: No representation of dialogue context Only the features concerning the FU Q and the answer A_0 (tfIdfSimilarityQA, lmProbQA) are used as predictors in building the logistic regression model. The result is a model that treats every FU Q as a stand-alone question. Across the five models, the coefficient for tfIdfSimilarityQA is roughly five times the size of that for lmProbQA.

Reduced model 2: No action sequences We keep only the two TF/IDF features (tfIdfSimilarityQA, tfIdfSimilarityAA). This model thus does not use any features that depend on human annotation, but only fully automatic features. The coefficient learned for tfIdfSimilarityQA is generally twice as large as that for tfIdfSimilarityAA.

Reduced model 3: No dialogue context, no action sequences Considered as a baseline, this model uses a single feature (tfIdfSimilarityQA) to predict answer correctness, favoring those answer candidates that have the highest lexical similarity wrt. the FU Q.

5 Discussion

In order to better understand the relatively high mean ranks of the correct answer candidates across Table 2, we scrutinized the results of the answer ranker (based on all tests on the "complete" model). The distribution of the ranks of correct answers is clearly skewed; in around half of the 76 cases, the correct answer was actually ranked among the top 20 of the 529 answer candidates. However, the mean correct rank deteriorates badly due to the lowest-ranking third of cases. Analyzing these lowest-ranking cases, it appears that they are often instances of two sub-classes of topic continuation FU Qs: (i) the FU Q is context-dependent, i.e., underspecified or exhibiting reference-related discourse phenomena; (ii) the FU Q is a slight variation of the previous question (e.g. only the wh-phrase changes, or only the object changes). This error analysis seems to suggest that it should be worthwhile to distinguish between sub-classes of topic-continuation FU Qs, and to improve specifically how answers for the "difficult" sub-classes are ranked.

The relatively high mean ranks are also due to the fact that in our approach of acquiring dialogue data, for each FU Q we marked only *one* answer from the whole repository as "correct". Again for the "complete" model, we checked the top 20 answer candidates that ranked higher than the actual "correct" one. We found that in over half of the cases an answer that could be considered correct was among the top 20.

Looking at the ranking results across the different models in Table 2, the fact that the "complete" model seems to outperform each of the three reduced models (although no statistical significance could be attained from comparing the rank numbers) confirms our two hypotheses proposed earlier. Firstly, identifying the underlying actions of questions/answers and modeling action-based sequences yield important information for identifying correct

	Reduced m. 3	**Reduced m. 2**	**Reduced m. 1**	**Complete model**
Predictors in model	tfIdfSimilarityQA	tfIdfSimilarityQA, tfIdfSimilarityAA	tfIdfSimilarityQA, lmProbQA	tfIdfSimilarityQA, tfIdfSimilarityAA, lmProbQA
Split 1	141.2	108.4	112.5	96.2
Split 2	102.7	97.4	53.8	57.7
Split 3	56.7	63.7	50.5	52.7
Split 4	40.5	26.2	37.9	35.7
Split 5	153.1	105.3	129.6	89.1
Mean	98.8	80.2	76.7	66.3

Table 2: Mean ranks of correct A_0 out of 529 answer candidates, across models and training/test splits

answers to topic continuation FU Qs. Secondly, as for the role of the immediate dialogue context for providing additional clues for identifying good answers to FU Qs, our data show that a high lexical similarity score between A_{-1} and A_0 indicates a correct answer candidate. While (Yang et al., 2006) point out the importance of Q_{-1} to provide context information, in our experiments it was generally superseded by A_{-1}.

As for the two features relating the underlying actions of A_{-1} and A_0 (actionContinuityAA, lmProbAA), the picture seems less clear; in our current modeling experiments, we had not enough evidence to keep these features. However, we plan to explore the underlying idea of action-action sequences in the future, and conjecture that such information should come into its own for *context-dependent* FU Qs.

6 Future work

Besides annotating and using more dialogue data as more people talk to our IQA system, we plan to implement a state-of-the-art topic-shift detection algorithm as proposed in (Yang et al., 2006), training and testing it on our own FU Q data. We will attempt to improve this system by adding action-based features, and then extend it to distinguish three classes: topic shifts, (topic continuation) FU Qs that are fully specified, and (topic continuation) context-dependent FU Qs. We then plan to build dedicated logistic regression models for the different sub-classes of topic continuation FU Qs. If each model uses a specific set of predictors, we hope to improve the overall rank of correct answers across the different classes of FU Qs. Also, from comparing the different models, we are interested in studying the specific properties of different FU Q types.

References

[Agresti2002] Alan Agresti. 2002. *Categorical Data Analysis*. Wiley-Interscience, New York.

[Baayen2008] R. Harald Baayen. 2008. *Analyzing Linguistic Data*. Cambridge University Press.

[Bertomeu et al.2006] Núria Bertomeu, Hans Uszkoreit, Anette Frank, Hans-Ulrich Krieger, and Brigitte Jörg. 2006. Contextual phenomena and thematic relations in database QA dialogues: results from a wizard-of-oz experiment. In *Proc. of the Interactive Question Answering Workshop at HLT-NAACL 2006*, pages 1–8, New York, NY.

[Dahlbäck and Jönsson1989] Nils Dahlbäck and Arne Jönsson. 1989. Empirical studies of discourse representations for natural language interfaces. In *Proc. of the 4th Conference of the European Chapter of the ACL (EACL'89)*, pages 291–298, Manchester, UK.

[Kirschner and Bernardi2007] Manuel Kirschner and Raffaella Bernardi. 2007. An empirical view on iqa follow-up questions. In *Proc. of the 8th SIGdial Workshop on Discourse and Dialogue*, Antwerp, Belgium.

[McCoy and Cheng1991] Kathleen F. McCoy and Jeannette Cheng. 1991. Focus of attention: Constraining what can be said next. In Cecile L. Paris, William R. Swartout, and William C. Mann, editors, *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, pages 103–124. Kluwer Academic Publishers, Norwell, MA.

[van Schooten et al.2009] Boris van Schooten, R. op den Akker, R. Rosset, O. Galibert, A. Max, and G. Illouz. 2009. Follow-up question handling in the IMIX and Ritel systems: A comparative study. *Journal of Natural Language Engineering*, 15(1):97–118.

[Yang et al.2006] Fan Yang, Junlan Feng, and Giuseppe Di Fabbrizio. 2006. A data driven approach to relevancy recognition for contextual question answering. In *Proc. of the Interactive Question Answering Workshop at HLT-NAACL 2006*, pages 33–40, New York City, NY.

Sentence Realisation from Bag of Words with dependency constraints

Karthik Gali, Sriram Venkatapathy
Language Technologies Research Centre,
IIIT-Hyderabad, Hyderabad, India
{karthikg@students,sriram@research}.iiit.ac.in

Abstract

In this paper, we present five models for sentence realisation from a bag-of-words containing minimal syntactic information. It has a large variety of applications ranging from Machine Translation to Dialogue systems. Our models employ simple and efficient techniques based on n-gram Language modeling.

We evaluated the models by comparing the synthesized sentences with reference sentences using the standard BLEU metric(Papineni et al., 2001). We obtained higher results (BLEU score of 0.8156) when compared to the state-of-art results. In future, we plan to incorporate our sentence realiser in Machine Translation and observe its effect on the translation accuracies.

1 Introduction

In applications such as Machine Translation (MT) and Dialogue Systems, sentence realisation is a major step. Sentence realisation involves generating a well-formed sentence from a bag of lexical items. These lexical items may be syntactically related to each other. The level of syntactic information attached to the lexical items might vary with application. In order to appeal to the wide range of applications that use sentence realisation, our experiments assume only basic syntactic information, such as unlabeled dependency relationships between the lexical items.

In this paper, we present different models for sentence realisation. These models consider a bag of words with unlabelled dependency relations as input and apply simple n-gram language modeling techniques to get a well-formed sentence.

We now present the role of a sentence realiser in the task of MT. In transfer-based approaches for MT[1] (Lavie et al., 2003), the source sentence is first analyzed by a parser (a phrase-structure or a dependency-based parser). Then the source lexical items are transferred to the target language using a bi-lingual dictionary. The target language sentence is finally realised by applying transfer-rules that map the grammar of both the languages. Generally, these transfer rules make use of rich analysis on the source side such as dependency labels etc. The accuracy of having such rich analysis (dependency labeling) is low and hence, might affect the performance of the sentence realiser. Also, the approach of manually constructing transfer rules is costly, especially for divergent language pairs such as English and Hindi or English and Japanese. Our models can be used in this scenario, providing a robust alternative to the transfer rules.

A sentence realiser can also be used in the framework of a two-step statistical machine translation. In the two-step framework, the semantic transfer and sentence realisation are decoupled into independent modules. This provides an opporunity to develop simple and efficient modules for each of the steps. The model for *Global Lexical Selection and Sentence Re-construction* (Bangalore et al., 2007) is one such approach. In this approach, discriminative techniques are used to first transfer semantic information of the source sentence by looking at the source sentence globally, this obtaining a accurate bag-of-words in the target language. The words in the bag might be attached with mild syntactic information (ie., the words they modify) (Venkatapathy and Bangalore, 2007). We propose models that take

[1] http://www.isi.edu/natural-language/mteval/html/412.html

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 19–24,
Boulder, Colorado, June 2009.

this information as input and produce the target sentence. We can also use our sentence realiser as an ordering module in other approaches such as (Quirk et al., 2005), where the goal is to order an unordered bag (of treelets in this case) with dependency links.

In Natural Language Generation applications such as Dialogue systems etc, the set of concepts and the dependencies between the concepts is obtained first which is known as text planning. These concepts are then realized into words resulting in a bag of words with syntactic relations (Bangalore and Rambow, 2000). This is known as sentence planning. In the end, the surface string can be obtained by our models.

In this paper, we do not test our models with any of the applications mentioned above. However, we plan to test our models with these applications, especially on the two-stage statistical MT approach using the bag-of-words obtained by Global Lexical Selection (Bangalore et al., 2007),(Venkatapathy and Bangalore, 2007). Here, we test our models independent of any application, by beginning with a given bag-of-words (with dependency links).

The structure of the paper is as follows. We give an overview of the related work in section 2. In section 3, we talk about the effect of dependency constraints and gives details of the experimental setup in section 4. In section 5, we describe about the experiments that have been conducted. In section 6, our experimental results are presented. In section 7, we talk about the possible future work and we conclude with section 8.

2 Related Work

There have been approaches for sentence realisation such as FUF/SURGE (Elhadad, 1991), OpenCCG (White, 2004) and XLE (Crouch et al., 2007) that apply hand-crafted grammars based on particular linguistic theories. These approaches expect rich syntactic information as input in order to realise the sentence. There are other approaches in which the generation grammars are extracted semi-automatically (Belz, 2007) or automatically (such as HPSG (Nakanishi and Miyao, 2005), LFG (Cahill and van Genabith, 2006; Hogan et al., 2007) and CCG (White et al., 2007)). The limitation of these approaches is that these cannot be incorporated into a wide range of applications as they rely on rich syntactic information for generation. On the contrary, we use simple n-gram models to realise (or linearize) a bag-of-words where the only information available is the presence of various links between the words.

Our work is similar to a recently published work by Guo (Guo et al., 2008). They use n-gram models to realise sentences from the f-structures of HPSG (equivalent to labeled dependency structure). Their models rely heavily on the dependency relation labels (also called grammatical roles) available in HPSG. However, the dependency role information (of any dependency formalism) is either not readily available in a variety of applications in NLP. We propose to explore the realisation of a sentence using minimal syntactic information. Apart from dependency links, we also make use of part-of-speech tags which are easily available and hence, our sentence realiser can be plugged much easily into various applications. Guo (Guo et al., 2008) conduct their experiments by considering gold data as input. Apart from using gold data as input, we also conduct experiments by assuming noisy input data to test the robustness of our models. The search algorithm used by both Guo and us is locally greedy i.e., we compute the best string at every node. Guo uses the Viterbi algorithm to get best string whereas we consider and score all permutations to obtain the best string.

There has been burgeoning interest in the probabilistic models for sentence realisation, especially for realisation ranking in a two stage sentence realisation architecture where in the first stage a set of sentence realisations are generated and then a realisation ranker will choose the best of them (Bangalore and Rambow, 2000).

One major observation in our experiments was that the POS tags held immensely in the task of sentence realisation.

3 Effect of Dependency Constraints

There is a major advantage in using dependency constraints for sentence realisation. The search space reduces drastically when the constraints are applied. These constraints state that the realised sentences should be projective with respect to the de-

pendency structure (unordered) of the input bag-of-words ie.., any word and its children in the dependency tree should project as a contiguous unit in the realised sentence. This is a safe assumption to make as the non-projectivity in English is only used to account for Long-Distance Dependencies and such cases are low in number (Guo et al., 2008).

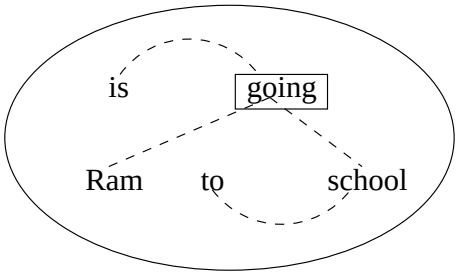

Figure 1: Bag of words with dependency constraints and head marked

We now present an example to show how the dependency constraints reduce the search space. For example, consider an unordered dependency tree in Figure 1, which has five words. If we don't use the constraints provided by the dependency tree then the search space is 5! (120). But, if we use the constraints provided by the dependency tree then the search space is 2! + 4! = 28. There is a huge reduction in the search space if we use the constraints provided by the dependency tree. Further, it has been shown in (Chang and Toutanova, 2007) that applying the constraints also aids for the synthesis of better constructed sentences.

4 Experimental Set-up

For the experiments, we use the WSJ portion of the Penn tree bank (Marcus et al., 1993), using the standard train/development/test splits, viz 39,832 sentences from 2-21 sections, 2416 sentences from section 23 for testing and 1,700 sentences from section 22 for development. The input to our sentence realiser are bag of words with dependency constraints which are automatically extracted from the Penn treebank using head percolation rules used in (Magerman, 1995), which do not contain any order information. We also use the provided part-of-speech tags in some experiments.

In a typical application, the input to the sentence realiser is noisy. To test the robustness of our models in such scenarios, we also conduct experiments with noisy input data. We parse the test data with an unlabelled projective dependency parser (Nivre et al., 2006) and drop the order information to obtain the input to our sentence realiser. However we still use the correct bag of words. We propose to test this aspect in future by plugging our sentence realiser in Machine Translation.

Table 1 shows the number of nodes having a particular number of children in the test data.

Children	countNodes		Children	countNodes
0	30219		5	1017
1	13649		6	685
2	5887		7	269
3	3207		8	106
4	1526		> 8	119

Table 1: The number of nodes having a particular number of children in the test data

From Table 1, we can see that more than 96% of the internal nodes of the trees contain five or less children. It means that for almost all the nodes, the reordering complexity is minimal. This makes this approach very feasible if the order of a sub-tree is computed after the order of the sub-trees of its children is fixed. Hence, the approaches that we present in the next section use bottom-up traversal of the tree. During the traversal, the appropriate order of every sub-tree is fixed.

5 Experiments

The task here is to realise a well formed sentence from a bag of words with dependency constraints (unordered dependency tree) for which we propose five models using n-gram based Language modeling techinque. We train the language models of order 3 using Good-Turning smoothing on the training data of Penn Treebank.

5.1 Model 1 : Sentential Language Model

We traverse the tree in bottom up manner and find the best phrase at each subtree. The best phrase corresponding to the subtree is assigned to the root node of the sub-tree during the traversal.

Let the node n have N children represented as c_i ($1 < i < N$). During the bottom up traversal, the

children c_i are assigned best phrases before processing node n. Let the best phrases corresponding to the children be $p(c_i)$. The best phrase corresponding to the node n is computed by exploring the permutations of n and the best phrases $p(c_i)$ corresponding to the children c_i. The total number of permutations that are explored are $(N+1)!$. A sentential language model is applied on each of the candidate phrases to select the best phrase.

$$p(n) = bestPhrase\ (\ perm\ (n,\ \forall\ i\ p(c_i))\ o\ LM\ ) \quad (1)$$

In Sentential Language Model, we used a LM that is trained on complete sentences of the training corpus to score the permutations.

5.2 Model 2 : Subtree-type based Language Models(STLM)

The major problem with model 1 is that we are using a common sentential language model (trained on complete sentences) to score phrases corresponding to various sub-tree types. In this model, we build different LMs for phrases corresponding to different subtree-types.

To build STLMs, the training data is parsed first. Each subtree in the parse structure is represented by the part-of-speech tag of its head. Different language models are created for each of the POS tags. We have 44 different language models each corresponding to a particular POS tag. For example, a *IN* language model contains phrases like *in hour, of chaos, after crash, in futures, etc* and VBD language model contains phrases like *were criticized, never resumed* while training.

So, in this model we realise a sentence from a unordered dependency tree by traversing the dependency tree in bottom-up manner as we did in model 1; but while scoring the permuted phrases we use different language models for subtrees headed by words of various pos tags.

$$p(n) = bestPhrase\ (\ perm\ (n,\ \forall\ i\ p(c_i))\ o\ LM_{POS(n)}\ ) \quad (2)$$

Here, $LM_{POS(n)}$ represents the language model associated with the part-of-speech of the node n.

5.3 Model 3 : Head-word STLM

In the models presented earlier, a node and its children are ordered using the best phrases of the children. For example, the best phrase assigned to the node 'was' is computed by taking of the permutation of 'was' and its children 'The equity market', 'illiquid' and '.' and then applying the language model. In model 3, instead of considering best phrases while ordering, the heads of the the children c_i are considered. For example, the best phrase assigned to the node 'was' is computed by first permuting the nodes 'was', 'market', 'illiquid' and '.' and then applying the language models trained on the treelets (head and children) and not on entire sub-trees.

The major advantage of using this model is that order at a node is independent of the best phrases of its descendants and also any mistakes in computation of best phrases of descendants doesn't effect the choice of reordering decision at a particular node.

5.4 Model 4 : POS based STLM

We now experiment by using Part-Of-Speech (POS) tags of words for ordering the nodes. In the previous approaches, the language models were trained on the words which were then used to compute the best strings associated with various nodes. Here, we order the node and its children using a language model trained on POS tag sequences. The motivation behind buliding such kind of Language models is that it deals with unseen words effectively. Hence, in this model, the best phrase corresponding to the node 'was' is obtained by permuting the POS tags of the words 'was', 'market', 'illiquid and '.' which are 'VBZ', 'NN', 'NN' and '.' respectively. As the best POS tag sequence might correspond to several orderings of the treelet, a word based STLM is applied to choose the correct ordering.

The major advantages of this model is that it is more general and it deals with unseen words effectively. Also, it is much faster than earlier models as this model is a POS tag based model.

5.5 Model 5: Head-marked POS based STLM

In POS based STLM, the head of a particular node isn't marked while applying the language model. Hence, all the nodes of the treelet are treated equally while applying the LM. For example, in Figure 2, the structures of treelets is not taken into account while applying the head-POS based language model. Both are treated in the same manner while applying TLM. In this model, we experiment by marking the head

information for the POS of the head word which treats the treelets in Figure 2 in a different manner to obtain the best phrase. As the best POS tag sequence might correspond to several orderings of the treelet, we test various word-based approaches to choose the best ordering among the many possibilities. The best approach was the one where head-word of the treelet had the POS tag attached to it.

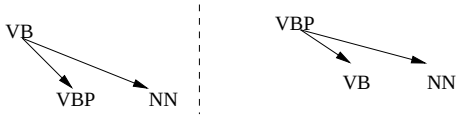

Figure 2: Two different treelets which would have same best POS tag sequence

6 Results and Discussion

To evaluate our models, we compare the system generated sentences with reference sentences and get the BLEU score. As mentioned in section 4, We evaluate our models on two different types of input. In the first input type, we have bag of words with dependency constraints extracted from treebank and in the second input type, the dependency constraints among the bag of words are extracted from the parser which are noisy. Table 2 shows the results of model 1-5.

Model	Treebank(gold)	Parser(noisy)
Model 1	0.5472	0.5514
Model 2	0.6886	0.6870
Model 3	0.7284	0.7227
Model 4	0.7890	0.7783
Model 5	0.8156	0.8027

Table 2: The results of Model 1-5

We can observe that in model 1, BLEU score of the parser input is high when compared to Treebank input. This might be because, the parser input is projective (as we used projective parsing) whereas the treebank input might contain some non-projective cases. In general, for all the models, the results with noisy dependency links are comparable to the cases where gold dependency links are used which is encouraging.

We have taken the Table-3 from (Guo et al., 2008), which shows the BLEU scores of different

Paper	BLEU score
Langkilde(2002)	0.757
Nakanishi(2005)	0.705
Cahill(2006)	0.6651
Hogan(2007)	0.6882
White(2007)	0.5768
Guo(2008)	0.7440
Our Model	0.8156

Table 3: Comparsion of results for English WSJ section 23

systems on section 23 of PTB. Its really difficult to compare sentence realisers as the information contained in the input vaires greatly between systems. But, we can clearly see that the our system performs better than all the systems. The main observations from the results are, (1) Searching the entire space of O(n!) helps, (2) Treelet LM capture characteristics of phrases headed by various POS tags, in contrast to sentential LM which is a general LM, (3) POS tags can play an important role in ordering nodes of a dependency structure, (4) The head models performed better than the models that used all the nodes of the sub-tree, and (5) Marking the head of a treelet provides vital clues to the language model for reordering.

7 Future Experiments

Although the results of the proposed models are much higher when compared to other methods, the major constraint with our models is the computational complexity, which is O(n!). However, our approach is still tractable because of the low values of n. We plan to reduce the search space complexity by using Viterbi search (Guo et al., 2008), and examine the drop in results because of that.

The models proposed in paper, consider only the locally best phrases (local to the sub-tree) at every step. In order to retain the globally best possibilities at every step, we plan to use beam search, where we retain K-best best phrases for every sub-tree.

Also, the goal is to test the approach for morphologically-rich languages such as Hindi. Also, it would require us to expand our features set. We also plan to test the factored models.

The most important experiment that we plan to

perform is to test our system in the context of MT, where the input is more real and noisy.

To train more robust language models, we plan to use the much larger data on a web scale.

8 Conclusion

In this paper, we had experimented with five ngram based models for sentence realisation from bag of words with dependency constraints. We have evaluated our models on two different types of input(gold and noisy). From the results, we can conclude that the model 'Marked Head-POS based LM' works best in both cases.

Acknowledgments

The authors of this work were supported by ILMT grant 11(10)/2006-HCC(TDIL) and EILMT grant 11(9)/2006HCC(TDIL). We would also like to thank the four reviewers for their valuable reviews.

References

S. Bangalore and O. Rambow. 2000. Exploiting a probabilistic hierarchical model for generation. *Proceedings of the 18th conference on Computational linguistics*.

S. Bangalore, P. Haffner, and S. Kanthak. 2007. Statistical Machine Translation through Global Lexical Selection and Sentence Reconstruction. In *Annual Meeting - ACL*, volume 45.

A. Belz. 2007. Probabilistic Generation of Weather Forecast Texts. In *Proceedings of NAACL HLT*.

A. Cahill and J. van Genabith. 2006. Robust PCFG-Based Generation Using Automatically Acquired LFG Approximations. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 44.

P.C. Chang and K. Toutanova. 2007. A Discriminative Syntactic Word Order Model for Machine Translation. *Proceedings of the 45th Annual Meeting of the ACL*.

D. Crouch, M. Dalrymple, R. Kaplan, T. King, J. Maxwell, and P. Newman. 2007. XLE documentation. *Available on-line*.

M. Elhadad. 1991. FUF: The universal unifier user manual version 5.0. *Department of Computer Science, Columbia University. New York*.

Y. Guo, J. van Genabith, and H. Wang. 2008. Dependency-Based N-Gram Models for General Purpose Sentence Realisation. *Proceedings of the 22nd conference on Computational linguistics*.

D. Hogan, C. Cafferkey, A. Cahill, and J. van Genabith. 2007. Exploiting Multi-Word Units in History-Based Probabilistic Generation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

A. Lavie, S. Vogel, L. Levin, E. Peterson, K. Probst, A.F. Llitjós, R. Reynolds, J. Carbonell, and R. Cohen. 2003. Experiments with a Hindi-to-English transfer-based MT system under a miserly data scenario. *ACM-TALIP*, 2(2).

D.M. Magerman. 1995. Statistical decision-tree models for parsing. In *Proceedings of the 33rd annual meeting on ACL*. ACL Morristown, NJ, USA.

M.P. Marcus, M.A. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of English: the penn treebank. *Computational Linguistics*, 19(2).

H. Nakanishi and Y. Miyao. 2005. Probabilistic models for disambiguation of an HPSG-based chart generator. In *Proceedings of the International Workshop on Parsing Technology*.

J. Nivre, J. Hall, J. Nilsson, G. Eryigit, and S. Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the Tenth CoNLL*.

K. Papineni, S. Roukos, T. Ward, and W.J. Zhu. 2001. BLEU: a method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting on ACL*.

C. Quirk, A. Menezes, and C. Cherry. 2005. Dependency treelet translation: syntactically informed phrasal SMT. *Proceedings of the 43rd Annual Meeting of ACL*.

S. Venkatapathy and S. Bangalore. 2007. Three models for discriminative machine translation using Global Lexical Selection and Sentence Reconstruction. In *Proceedings of SSST, NAACLHLT/AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 152–159.

M. White, R. Rajkumar, and S. Martin. 2007. Towards Broad Coverage Surface Realization with CCG. In *Proceedings of the Workshop on Using Corpora for NLG: Language Generation and Machine Translation (UCNLG+ MT)*.

M. White. 2004. Reining in CCG Chart Realization. *LECTURE NOTES IN COMPUTER SCIENCE*.

Using Language Modeling to Select Useful Annotation Data

Dmitriy Dligach
Department of Computer Science
University of Colorado at Boulder
Dmitriy.Dligach@colorado.edu

Martha Palmer
Department of Linguistics
University of Colorado at Boulder
Martha.Palmer@colorado.edu

Abstract

An annotation project typically has an abundant supply of unlabeled data that can be drawn from some corpus, but because the labeling process is expensive, it is helpful to pre-screen the pool of the candidate instances based on some criterion of future usefulness. In many cases, that criterion is to improve the presence of the rare classes in the data to be annotated. We propose a novel method for solving this problem and show that it compares favorably to a random sampling baseline and a clustering algorithm.

1 Introduction

A data set is imbalanced when the distribution of classes in it is dominated by a single class. In Word Sense Disambiguation (WSD), the classes are word senses. The problem of imbalanced data is painfully familiar to WSD researchers: word senses are particularly well known for their skewed distributions that are also highly domain and corpus dependent. Most polysemous words have a sense that occurs in a disproportionately high number of cases and another sense that is seen very infrequently. For example, the OntoNotes (Hovy et al., 2006) sense inventory defines two senses for the verb *to add*. Of all the instances of this verb in the OntoNotes sense-tagged corpus, 93% are the instances of the predominant sense (not the arithmetic sense!). Another fact: there are 4,554 total senses in the OntoNotes sense inventory for 1,713 recently released verbs. Only 3,498 of them are present in the actual annotated data. More than 1,000 senses (23%) are so rare that they are missing from the corpus altogether. More than a third of the released verbs are missing representative instances of at least one sense. In fact many of the verbs are pseudo-monosemous: even though the sense inventory defines multiple senses, only the most frequent sense is present in the actual annotated data. For example, only 1 out of 8 senses of *to rip* is present in the data.

The skewed nature of sense distributions is a fact of life. At the same time, a large-scale annotation project like OntoNotes, whose goal is the creation of a comprehensive linguistic resource, cannot simply ignore it. That a sense is rare in a corpus does not mean that it is less important to annotate a sufficient number of instances of that sense: in a different domain it can be more common and not having enough annotated instances of that sense could jeopardize the success of an automatic cross-domain WSD system. For example, sense 8 of *to rip* ("to import an audio file directly from CD") is extremely popular on the web but it does not exist at all in the OntoNotes data. Only the traditional sense of *to swap* exists in the data but not the computer science sense ("to move a piece of program into memory"), while the latter can conceivably be significantly more popular in technical domains.

In general, class imbalance complicates supervised learning. This contention certainly holds for WSD. As an illustration, consider the verb *to call*, for which the OntoNotes sense inventory defines 11 senses. Senses 3 and 5 are the most frequent: together they constitute 84% of the data. To investigate which classes are problematic for a classifi-

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 25–30,
Boulder, Colorado, June 2009.

er, we conducted 50 supervised learning experiments. In each experiment one instance of this verb was selected at random and used for testing while the rest was used for training a maximum entropy model. The resulting confusion matrix shows that the model correctly classified most of the instances of the two predominant senses while misclassifying the other classes. The vast majority of the errors came from confusing other senses with sense 5 which is the most frequent sense of *to call*. Clearly, the data imbalance problem has a significant negative effect on performance.

Let us now envision the following realistic scenario: An annotation project receives funds to sense-tag a set of verbs in a corpus. It may be the case that some annotated data is already available for these verbs and the goal is to improve sense coverage, or no annotated data is available at all. But it turns out there are only enough funds to annotate a portion (e.g. half) of the total instances. The question arises how to pre-select the instances from the corpus in a way that would ensure that all the senses are as well represented as possible. Because some senses of these verbs are very rare, the pool of instances pre-selected for the annotation should include as many as possible instances of the rare senses. Random sampling – the simplest approach – will clearly not work: the pre-selected data will contain roughly the same proportion of the rare sense instances as the original set.

If random sampling is not the answer, the data must be selected in some non-uniform way, i.e. using *selective* sampling. Active learning (e.g. Chen et al., 2006) is one approach to this problem. Some evidence is available (Zhu and Hovy, 2007) that active learning outperforms random sampling in finding the instances of rare senses. However, active learning has several shortcomings: (1) it requires some annotated data to start the process; (2) it is problematic when the initial training set only contains the data for a single class (e.g. the pseudo-monosemous verbs); (3) it is not always efficient in practice: In the OntoNotes project, the data is annotated by two human taggers and the disagreements are adjudicated by the third. In classic active learning a single instance is labeled on each iteration This means the human taggers would have to wait on each other to tag the instance, on the adjudicator for the resolution of a possible disagreement, and finally on the system which still needs to be-retrained to select the next instance to be labeled, a time sink much greater than tagging additional instances; (4) finally, active learning may not be an option if the data selected needs to be manually pre-processed (e.g. sentence segmented, tokenized, and treebanked – as was the case with some of the OntoNotes data). In this setting, on each iteration of the algorithm, the taggers have to also wait for the selected instance to be manually pre-processed before they can label it.

Thus, it would be significantly more convenient if all the data to be annotated could be pre-selected **in advance**. In this paper we turn to two unsupervised methods which have the potential to achieve that goal. We propose a simple language modeling-based sampling method (abbreviated as **LMS**) that increases the likelihood of seeing rare senses in the pre-selected data. The basic approach is as follow: using language modeling we can rank the instances of the ambiguous verb according to their probability of occurrence in the corpus. Because the instances of the rare senses are less frequent than the instances of the predominant sense, we can expect that there will be a higher than usual concentration of the rare sense instances among the instances that have low probabilities. The method is completely unsupervised and the only resource that it requires is a Language Modeling toolkit such as SRILM (Stolcke, 2002), which we used in our experiments. We compare this method with a random sampling baseline and semi-supervised clustering, which can serve the same purpose. We show that our method outperforms both of the competing approaches. We review the relevant literature in section 2, explain the details of LMS in section 3, evaluate LMS in section 4, discuss the results in section 5, and describe our plans for future work in section 6.

2 Relevant Work

The problem of imbalanced data has recently received much attention in the machine learning community. Rare classes can be of higher importance than frequent classes, as in medical diagnosis when one is interested in correctly identifying a rare disease. Network intrusion detection faces a similar problem: a malicious activity, although of crucial importance, is a very rare event compared to the large volumes of routine network traffic. At the same time, imbalanced data poses difficulties for an automatic learner in that rare classes have a much higher misclassification rate than common

ones (Weiss, 1995; Japkowicz, 2001). Learning from imbalanced sets can also be problematic if the data is noisy: given a sufficiently high level of background noise, a learner may not distinguish between true exceptions (i.e. rare cases) and noise (Kubat and Matwin, 1997; Weiss, 2004).

In the realm of supervised learning, cost-sensitive learning has been recommended as a solution to the problem of learning from imbalanced data (e.g. Weiss, 2004). However, the costs of misclassifying the senses are highly domain specific and hard to estimate. Several studies recently appeared that attempted to apply active learning principles to rare category detection (Pelleg and Moore, 2004; He and Carbonell, 2007). In addition to the issues with active learning outlined in the introduction, the algorithm described in (He and Carbonell, 2007) requires the knowledge of the priors, which is hard to obtain for word senses.

WSD has a long history of experiments with unsupervised learning (e.g. Schutze, 1998; Purandare and Peterson, 2004). McCarthy et al. (2004) propose a method for automatically identifying the predominant sense in a given domain. Erk (2006) describes an application of an outlier detection algorithm to the task of identifying the instances of unknown senses. Our task differs from the latter two works in that it is aimed at finding the instances of the rare senses.

Finally, the idea of LMS is similar to the techniques for sentence selection based on rare n-gram co-occurrences used in machine translation (Eck et al., 2005) and syntactic parsing (Hwa, 2004).

3 Language Modeling for Data Selection

Our method is outlined in Figure 1:

Input
A large corpus that contains *T* candidate instances from which *S* instances are to be selected for annotation
Basic Steps
1. Compute the language model for the corpus 2. Compute the probability distribution over the *T* candidate instances of the target verb 3. Rank the *T* candidate instances by their probabilities 4. Form a cluster by selecting *S* instances with the lowest probability

Figure 1. Basic steps of LMS

Let us now clarify a few practical points. Although an instance of the target verb can be represented as the entire sentence containing the verb, from the experiments with automatic WSD (e.g. Dligach and Palmer, 2008), it is known that having access to just a few words in the neighborhood of the target verb is sufficient in many cases to predict the sense. For the purpose of LMS we represent an instance as the chunk of text centered upon the target verb plus the surrounding words on both sides within a three-word window. Although the size of the window around the target verb is fixed, the actual number of words in each chunk may vary when the target verb is close to the beginning or the end of sentence. Therefore, we need some form of length normalization. We normalize the log probability of each chunk by the actual number of words to make sure we do not favor shorter chunks (SRILM operates in log space). The resulting metric is related to perplexity: for a sequence of words $W = w_1 w_2 \ldots w_N$ the perplexity is

$$PP(W) = P(w_1 w_2 ... w_N)^{-\frac{1}{N}}$$

The log of perplexity is

$$\log[PP(W)] = -\frac{1}{N}\log[P(w_1 w_2 ... w_N)]$$

Thus, the quantity we use for ranking is negative perplexity.

4 Evaluation

For the evaluation, we selected two-sense verbs from the OntoNotes data that have at least 100 instances and where the share of the rare sense is less than 20%. There were 11 such verbs (2,230 instances total) with the average share of the rare sense 11%.

Our task consists of clustering the instances of a verb into two clusters, one of which is expected to have a higher concentration of the rare senses than the other. Since the rare sense cluster is of primary interest to us, we report two metrics: (1) precision: the ratio of the number of instances of the rare sense in the cluster and the total number of instances in the cluster; (2) recall: the ratio of the number of instances of the rare sense in the cluster and the total number of the rare sense instances in both clusters. Note that precision is not of primary importance for this task because the goal is not to reliably identify the instances of the rare sense but

rather to group them into a cluster where the rare senses will have a higher concentration than in the original set of the candidate instances. At the same time achieving high recall is important since we want to ensure that most, if not all, of the rare senses that were present among the candidate instances are captured in the rare sense cluster.

4.1 Plausibility of LMS

The goal of our first set of experiments is to illustrate the plausibility of LMS. Due to space constraints, we examine only two verbs: *compare* and *add*. The remaining experiments will focus on a more comprehensive evaluation that will involve all 11 verbs. We computed the normalized log probability for each instance of a verb. We then ordered these candidate instances by their normalized log probability and computed the recall of the rare sense at various levels of the size of the rare sense cluster. We express the size of the rare sense cluster as a share of the total number of instances. We depict recall vs. cluster size with a dotted curve. The graphs are in Figures 2 and 3.

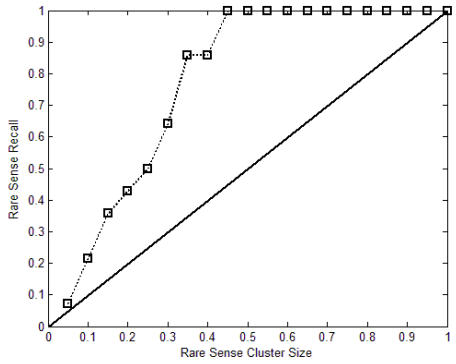

Figure 2. Rare sense recall for *compare*

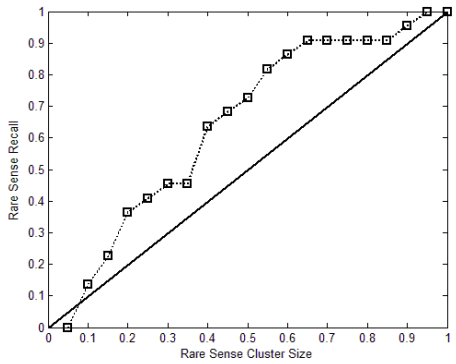

Figure 3. Rare sense recall for *add*

The diagonal line on these figures corresponds to the random sampling baseline. A successful LMS would correspond to the dotted curve lying above the random sampling baseline, which happens to be the case for both of these verbs. For *compare* we can capture all of the rare sense instances in a cluster containing less than half of the candidate instances. While verbs like *compare* reflect the best-case scenario, the technique we proposed still works for the other verbs although not always as well. For example, for *add* we can recall more than 70% of the rare sense instances in a cluster that contains only half of all instances. This is more than 20 percentage points better than the random sampling baseline where the recall of the rare sense instances would be approximately 50%.

4.2 LMS vs. Random Sampling Baseline

In this experiment we evaluated the performance of LMS for all 11 verbs. For each verb, we ranked the instances by their normalized log probability and placed the bottom half in the rare sense cluster. The results are in Table 2. The second column shows the share of the rare sense instances in the entire corpus for each verb. Thus, it represents the precision that would be obtained by random sampling. The recall for random sampling in this setting would be 0.5.

Ten verbs outperformed the random sampling baseline both with respect to precision and recall (although recall is much more important for this task) and one verb performed as well. On average these verbs showed a recall figure that was 22 percentage points better than random sampling. Two of the 11 verbs (*compare* and *point*) were able to recall all of the rare sense instances.

Verb	Rare Inst	Precision	Recall
account	0.12	0.21	0.93
add	0.07	0.10	0.73
admit	0.18	0.18	0.50
allow	0.06	0.07	0.62
compare	0.08	0.16	1.00
explain	0.10	0.12	0.60
maintain	0.11	0.11	0.53
point	0.15	0.29	1.00
receive	0.07	0.08	0.60
remain	0.15	0.20	0.65
worry	0.15	0.22	0.73
average	0.11	0.16	0.72

Table 2. LMS results for 11 verbs

4.3 LMS vs. K-means Clustering

Since LMS is a form of clustering one way to evaluate its performance is by comparing it with an established clustering algorithm such as K-means (Hastie et al., 2001). There are several issues related to this evaluation. First, K-means produces clusters and which cluster represents which class is a moot question. Since for the purpose of the evaluation we need to know which cluster is most closely associated with a rare sense, we turn K-means into a semi-supervised algorithm by seeding the clusters. This puts LMS at a slight disadvantage since LMS is a completely unsupervised algorithm, while the new version of K-means will require an annotated instance of each sense. However, this disadvantage is not very significant: in a real-world application, the examples from a dictionary can be used to seed the clusters. For the purpose of this experiment, we simulated the examples from a dictionary by simply taking the seeds from the pool of the annotated instances we identified for the evaluation. K-means is known to be highly sensitive to the choice of the initial seeds. Therefore, to make the comparison fair, we perform the clustering ten times and pick the seeds at random for each iteration. The results are averaged.

Second, K-means generates clusters of a fixed size while the size of the LMS-produced clusters can be easily varied. This advantage of the LMS method has to be sacrificed to compare its performance to K-means. We compare LMS to K-means by counting the number of instances that K-means placed in the cluster that represents the rare sense and selecting the same number of instances that have the lowest normalized probability. Thus, we end up with the two methods producing clusters of the same size (with k-means dictating the cluster size).

Third, K-means operates on vectors and therefore the instances of the target verb need to be represented as vectors. We replicate lexical, syntactic, and semantic features from a verb sense disambiguation system that showed state-of-the-art performance on the OntoNotes data (Dligach and Palmer, 2008).

The results of the performance comparison are shown in Table 3. The fourth column shows the relative size of the K-means cluster that was seeded with the rare sense. Therefore it also defines the share of the instances with the lowest normalized log probability that are to be included in the LMS-produced rare sense clusters. On average, LMS showed 3% better recall than K-means clustering.

	K-means			LMS	
verb	precision	recall	size	precision	recall
account	0.21	1.00	0.58	0.20	1.00
add	0.06	0.54	0.50	0.10	0.73
admit	0.21	0.31	0.29	0.09	0.15
allow	0.08	0.36	0.31	0.06	0.31
compare	0.22	0.42	0.18	0.19	0.43
explain	0.16	0.61	0.44	0.14	0.60
maintain	0.13	0.91	0.80	0.11	0.82
point	0.27	0.66	0.42	0.31	0.89
receive	0.11	0.68	0.72	0.08	0.80
remain	0.10	0.41	0.44	0.21	0.61
worry	0.81	0.51	0.13	0.38	0.33
average	0.21	0.58	0.44	0.17	0.61

Table 3. LMS vs. K-means

5 Discussion and Conclusion

In this paper we proposed a novel method we termed LMS for pre-selecting instances for annotation. This method is based on computing the probability distribution over the instances and selecting the ones that have the lowest probability. The expectation is that instances selected in this fashion will capture more of the instances of the rare classes than would have been captured by random sampling. We evaluated LMS by comparing it to random sampling and showed that LMS outperforms it. We also demonstrated that LMS compares favorably to K-means clustering. This is despite the fact that the cluster sizes were dictated by K-means and that K-means had at its disposal much richer linguistic representations and some annotated data.

Thus, we conclude that LMS is a promising method for data selection. It is simple to use since one only needs the basic functionality that any language modeling toolkit offers. It is flexible in that the number of the instances to be selected can be specified by the user, unlike, for example, when clustering using k-means.

6 Future Work

First, we would like to investigate the effect of selective sampling methods (including LMS) on the performance of WSD models learned from the selected data. Next, we plan to apply LMS for Domain adaptation. Unlike the scenario we dealt with in this paper, the language model would have to be learned from and applied to different corpora: it would be trained on the source corpus and used to compute probabilities for the instances in the target corpus that needs to be adapted. We will also experiment with various outlier detection techniques to determine their applicability to data selection. Another promising direction is a simplified active learning approach in which a classifier is trained on the labeled data and applied to unlabeled data; the instances with a low classifier's confidence are selected for annotation (i.e. this is active learning conducted over a single iteration). This approach is more practical than the standard active learning for the reasons mentioned in Section 1 and should be compared to LMS. Finally, we will explore the utility of LMS-selected data as the initial training set for active learning (especially in the cases of the pseudo-monosemous verbs).

Acknowledgments

We gratefully acknowledge the support of the National Science Foundation Grant NSF-0715078, Consistent Criteria for Word Sense Disambiguation, and the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022, a subcontract from the BBN-AGILE Team. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Jinying Chen, Andrew Schein, Lyle Ungar, and Martha Palmer. 2006. An Empirical Study of the Behavior of Active Learning for Word Sense Disambiguation. In Proceedings of the HLT-NAACL.

Dmitriy Dligach and Martha Palmer. 2008. Novel Semantic Features for Verb Sense Disambiguation. In Proceedings of ACL-HLT.

Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Low Cost Portability for Statistical Machine Translation Based on N-gram Frequency and TF-IDF. Proceedings of IWSLT 2005.

Katrin Erk. Unknown Word Sense Detection as Outlier Detection. 2006. In Proceedings of HLT-NAACL.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The Elements of Statistical Learning. Data Mining, Inference, and Prediction. 2001. Springer.

Jingrui He and Jaime Carbonell. 2007. Nearest-Neighbor-Based Active Learning for Rare Category Detection. NIPS.

Hovy, E.H., M. Marcus, M. Palmer, S. Pradhan, L. Ramshaw, and R. Weischedel. 2006. OntoNotes: The 90% Solution. In Proceedings of the HLT-NAACL.

Eduard Hovy and Jingbo Zhu. 2007. Active Learning for Word Sense Disambiguation with Methods for Addressing the Class Imbalance Problem. In Proceedings of EMNLP.

Rebecca Hwa. 2004. Sample Selection for Statistical Parsing. Computational Linguistics. Volume 30. Issue 3.

Natalie Japkowicz. 2001. Concept Learning in the Presence of Between-Class and Within-Class Imbalances. Proceedings of the Fourteenth Conference of the Canadian Society for Computational Studies of Intelligence, Springer-Verlag.

Miroslav Kubat and Stan Matwin. 1997. Addressing the curse of imbalanced training sets: one-sided selection. In Proceedings of the Fourteenth International Conference on Machine Learning.

Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2004. Finding Predominant Word Senses in Untagged Text. In Proceedings of 42nd Annual Meeting of Association for Computational Linguistics.

Dan Pelleg and Andrew Moore. 2004. Active Learning for Anomaly and Rare-Category Detection. NIPS.

Amruta Purandare and Ted Pedersen. Word Sense Discrimination by Clustering Contexts in Vector and Similarity Spaces. 2004. In Proceedings of the Conference on CoNLL.

Hinrich Schutze. 1998 Automatic Word Sense Discrimination. Computational Linguistics.

Andreas Stolcke. 2002. SRILM - An Extensible Language Modeling Toolkit. In Proc. Intl. Conf. Spoken Language Processing, Denver, Colorado.

Gary M. Weiss. 1995. Learning with Rare Cases and Small Disjuncts. Proceedings of the Twelfth International Conference on Machine Learning, Morgan Kaufmann.

Gary M. Weiss. 2004. Mining with Rarity: A Unifying Framework. SIGKDD Explorations, special issue on learning from imbalanced datasets.

Pronunciation Modeling in Spelling Correction for Writers of English as a Foreign Language

Adriane Boyd
Department of Linguistics
The Ohio State University
1712 Neil Avenue
Columbus, Ohio 43210, USA
adriane@ling.osu.edu

Abstract

We propose a method for modeling pronunciation variation in the context of spell checking for non-native writers of English. Spell checkers, typically developed for native speakers, fail to address many of the types of spelling errors peculiar to non-native speakers, especially those errors influenced by differences in phonology. Our model of pronunciation variation is used to extend a pronouncing dictionary for use in the spelling correction algorithm developed by Toutanova and Moore (2002), which includes models for both orthography and pronunciation. The pronunciation variation modeling is shown to improve performance for misspellings produced by Japanese writers of English.

1 Introduction

Spell checkers identify misspellings, select appropriate words as suggested corrections, and rank the suggested corrections so that the likely intended word is high in the list. Since traditional spell checkers have been developed with competent native speakers as the target users, they do not appropriately address many types of errors made by non-native writers and they often fail to suggest the appropriate corrections. Non-native writers of English struggle with many of the same idiosyncrasies of English spelling that cause difficulty for native speakers, but differences between English phonology and the phonology of their native language lead to types of spelling errors not anticipated by traditional spell checkers (Okada, 2004; Mitton and Okada, 2007).

Okada (2004) and Mitton and Okada (2007) investigate spelling errors made by Japanese writers of English as a foreign language (JWEFL). Okada (2004) identifies two main sources of errors for JWEFL: differences between English and Japanese phonology and differences between the English alphabet and the Japanese *romazi* writing system, which uses a subset of English letters. Phonological differences result in number of distinctions in English that are not present in Japanese and *romazi* causes difficulties for JWEFL because the Latin letters correspond to very different sounds in Japanese.

We propose a method for creating a model of pronunciation variation from a phonetically untranscribed corpus of read speech recorded by non-native speakers. The pronunciation variation model is used to generate multiple pronunciations for each canonical pronunciation in a pronouncing dictionary and these variations are used in the spelling correction approach developed by Toutanova and Moore (2002), which uses statistical models of spelling errors that consider both orthography and pronunciation. Several conventions are used throughout this paper: a *word* is a sequence of characters from the given alphabet found in the word list. A *word list* is a list of words. A *misspelling*, marked with $\star$, is a sequence of characters not found in the word list. A *candidate correction* is a word from the word list proposed as a potential correction.

2 Background

Research in spell checking (see Kukich, 1992, for a survey of spell checking research) has focused on three main problems: non-word error detection, isolated-word error correction, and context-dependent word correction. We focus on the first two tasks. A non-word is a sequence of letters that

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 31–36,
Boulder, Colorado, June 2009.

is not a possible word in the language in any context, e.g., English *thier. Once a sequence of letters has been determined to be a non-word, isolated-word error correction is the process of determining the appropriate word to substitute for the non-word.

Given a sequence of letters, there are thus two main subtasks: 1) determine whether this is a non-word, 2) if so, select and rank candidate words as potential corrections to present to the writer. The first subtask can be accomplished by searching for the sequence of letters in a word list. The second subtask can be stated as follows (Brill and Moore, 2000): Given an alphabet Σ, a word list D of strings $\in \Sigma^*$, and a string $r \notin D$ and $\in \Sigma^*$, find $w \in D$ such that w is the most likely correction. Minimum edit distance is used to select the most likely candidate corrections. The general idea is that a minimum number of edit operations such as insertion and substitution are needed to convert the misspelling into a word. Words requiring the smallest numbers of edit operations are selected as the candidate corrections.

2.1 Edit Operations and Edit Weights

In recent spelling correction approaches, edit operations have been extended beyond single character edits and the methods for calculating edit operation weights have become more sophisticated. The spelling error model proposed by Brill and Moore (2000) allows generic string edit operations up to a certain length. Each edit operation also has an associated probability that improves the ranking of candidate corrections by modeling how likely particular edits are. Brill and Moore (2000) estimate the probability of each edit from a corpus of spelling errors. Toutanova and Moore (2002) extend Brill and Moore (2000) to consider edits over both letter sequences and sequences of phones in the pronunciations of the word and misspelling. They show that including pronunciation information improves performance as compared to Brill and Moore (2000).

2.2 Noisy Channel Spelling Correction

The spelling correction models from Brill and Moore (2000) and Toutanova and Moore (2002) use the noisy channel model approach to determine the types and weights of edit operations. The idea behind this approach is that a writer starts out with the intended word w in mind, but as it is being written the word passes through a noisy channel resulting in the observed non-word r. In order to determine how likely a candidate correction is, the spelling correction model determines the probability that the word w was the intended word given the misspelling r: $P(w|r)$. To find the best correction, the word w is found for which $P(w|r)$ is maximized: $argmax_w\ P(w|r)$. Applying Bayes' Rule and discarding the normalizing constant $P(r)$ gives the correction model:

$$argmax_w\ P(w|r) = argmax_w\ P(w)P(r|w)$$

$P(w)$, how probable the word w is overall, and $P(r|w)$, how probable it is for a writer intending to write w to output r, can be estimated from corpora containing misspellings. In the following experiments, $P(w)$ is assumed be equal for all words to focus this work on estimating the error model $P(r|w)$ for JWEFL.[1]

Brill and Moore (2000) allow all edit operations $\alpha \rightarrow \beta$ where Σ is the alphabet and $\alpha, \beta \in \Sigma^*$, with a constraint on the length of α and β. In order to consider all ways that a word w may generate r with the possibility that any, possibly empty, substring α of w becomes any, possibly empty, substring β of r, it is necessary to consider all ways that w and r may be partitioned into substrings. This error model over letters, called P_L, is approximated by Brill and Moore (2000) as shown in Figure 1 by considering only the pair of partitions of w and r with the maximum product of the probabilities of individual substitutions. $Part(w)$ is all possible partitions of w, $|R|$ is number of segments in a particular partition, and R_i is the i^{th} segment of the partition.

The parameters for $P_L(r|w)$ are estimated from a corpus of pairs of misspellings and target words. The method, which is described in detail in Brill and Moore (2000), involves aligning the letters in pairs of words and misspellings, expanding each alignment with up to N neighboring alignments, and calculating the probability of each $\alpha \rightarrow \beta$ alignment. Since we will be using a training corpus that consists solely of pairs of misspellings and words (see section 3), we would have lower probabilities for

[1] Of course, $P(w)$ is not equal for all words, but it is not possible to estimate it from the available training corpus, the Atsuo-Henry Corpus (Okada, 2004), because it contains only pairs of words and misspellings for around 1,000 target words.

$$P_L(r|w) \approx max_{R \in Part(r), T \in Part(w)} \prod_{i=1}^{|R|} P(R_i \rightarrow T_i)$$

$$P_{PHL}(r|w) \approx \sum_{pron_w} \frac{1}{|pron_w|} \max_{pron_r} P_{PH}(pron_w|pron_r) P(pron_r|r)$$

Figure 1: Approximations of P_L from Brill and Moore (2000) and P_{PHL} from Toutanova and Moore (2002)

$\alpha \rightarrow \alpha$ than would be found in a corpus with misspellings observed in context with correct words. To compensate, we approximate $P(\alpha \rightarrow \alpha)$ by assigning it a minimum probability m:

$$P(\alpha \rightarrow \beta) = \begin{cases} m + (1-m)\frac{count(\alpha \rightarrow \beta)}{count(\alpha)} & \text{if } \alpha = \beta \\ (1-m)\frac{count(\alpha \rightarrow \beta)}{count(\alpha)} & \text{if } \alpha \neq \beta \end{cases}$$

2.2.1 Extending to Pronunciation

Toutanova and Moore (2002) describe an extension to Brill and Moore (2000) where the same noisy channel error model is used to model phone sequences instead of letter sequences. Instead of the word w and the non-word r, the error model considers the pronunciation of the non-word r, $pron_r$, and the pronunciation of the word w, $pron_w$. The error model over phone sequences, called P_{PH}, is just like P_L shown in Figure 1 except that r and w are replaced with their pronunciations. The model is trained like P_L using alignments between phones.

Since a spelling correction model needs to rank candidate words rather than candidate pronunciations, Toutanova and Moore (2002) derive an error model that determines the probability that a word w was spelled as the non-word r based on their pronunciations. Their approximation of this model, called P_{PHL}, is also shown in Figure 1. $P_{PH}(pron_w|pron_r)$ is the phone error model described above and $P(pron_r|r)$ is provided by the letter-to-phone model described below.

2.3 Letter-To-Phone Model

A letter-to-phone (LTP) model is needed to predict the pronunciation of misspellings for P_{PHL}, since they are not found in a pronouncing dictionary. Like Toutanova and Moore (2002), we use the n-gram LTP model from Fisher (1999) to predict these pronunciations. The n-gram LTP model predicts the pronunciation of each letter in a word considering up to four letters of context to the left and right. The most specific context found for each letter and its context in the training data is used to predict the pronunciation of a word. We extended the prediction step to consider the most probable phone for the top M most specific contexts.

We implemented the LTP algorithm and trained and evaluated it using pronunciations from CMUDICT. A training corpus was created by pairing the words from the size 70 CMUDICT-filtered SCOWL word list (see section 3) with their pronunciations. This list of approximately 62,000 words was split into a training set with 80% of entries and a test set with the remaining 20%. We found that the best performance is seen when $M = 3$, giving 95.5% phone accuracy and 74.9% word accuracy.

2.4 Calculating Final Scores

For a misspelling r and a candidate correction w, the letter model P_L gives the probability that w was written as r due to the noisy channel taking into account only the orthography. P_{PH} does the same for the pronunciations of r and w, giving the probability that $pron_w$ was output was $pron_r$. The pronunciation model P_{PHL} relates the pronunciations modeled by P_{PH} to the orthography in order to give the probability that r was written as w based on pronunciation. P_L and P_{PHL} are then combined as follows to calculate a score for each candidate correction.

$$S_{CMB}(r|w) = logP_L(r|w) + \lambda logP_{PHL}(r|w)$$

3 Resources and Data Preparation

Our spelling correction approach, which includes error models for both orthography and pronunciation (see section 2.2) and which considers pronunciation variation for JWEFL requires a number of resources: 1) spoken corpora of American English (TIMIT, TIMIT 1991) and Japanese English (ERJ, see below) are used to model pronunciation variation, 2) a pronunciation dictionary (CMUDICT, CMUDICT 1998) provides American English pronunciations for the target words, 3) a corpus of

spelling errors made by JWEFL (Atsuo-Henry Corpus, see below) is used to train spelling error models and test the spell checker's performance, and 4) Spell Checker Oriented Word Lists (SCOWL, see below) are adapted for our use.

The **English Read by Japanese Corpus** (Minematsu et al., 2002) consists of 70,000 prompts containing phonemic and prosodic cues recorded by 200 native Japanese speakers with varying English competence. See Minematsu et al. (2002) for details on the construction of the corpus.

The **Atsuo-Henry Corpus** (Okada, 2004) includes a corpus of spelling errors made by JWEFL that consists of a collection of spelling errors from multiple corpora.[2] For use with our spell checker, the corpus has been cleaned up and modified to fit our task, resulting in 4,769 unique misspellings of 1,046 target words. The data is divided into training (80%), development (10%), and test (10%) sets.

For our word lists, we use adapted versions of the **Spell Checker Oriented Word Lists**.[3] The size 50 word lists are used in order to create a general purpose word list that covers all the target words from the Atsuo-Henry Corpus. Since the target pronunciation of each item is needed for the pronunciation model, the word list was filtered to remove words whose pronunciation is not in CMUDICT. After filtering, the word list contains 54,001 words.

4 Method

This section presents our method for modeling pronunciation variation from a phonetically untranscribed corpus of read speech. The pronunciation-based spelling correction approach developed in Toutanova and Moore (2002) requires a list of possible pronunciations in order to compare the pronunciation of the misspelling to the pronunciation of correct words. To account for target pronunciations specific to Japanese speakers, we observe the pronunciation variation in the ERJ and generate additional pronunciations for each word in the word list. Since the ERJ is not transcribed, we begin by adapting a recognizer trained on native English speech. First, the ERJ is recognized using a monophone recognizer trained on TIMIT. Next, the most frequent variations between the canonical and recognized pronunciations are used to adapt the recognizer. The adapted recognizer is then used to recognize the ERJ in forced alignment with the canonical pronunciations. Finally, the variations from the previous step are used to create models of pronunciation variation for each phone, which are used to generate multiple pronunciations for each word.

4.1 Initial Recognizer

A monophone speech recognizer was trained on all TIMIT data using the Hidden Markov Model Toolkit (HTK).[4] This recognizer is used to generate a phone string for each utterance in the ERJ. Each recognized phone string is then aligned with the canonical pronunciation provided to the speakers. Correct alignments and substitutions are considered with no context and insertions are conditioned on the previous phone. Due to restrictions in HTK, deletions are currently ignored.

The frequency of phone alignments for all utterances in the ERJ are calculated. Because of the low phone accuracy of monophone recognizers, especially on non-native speech, alignments are observed between nearly all pairs of phones. In order to focus on the most frequent alignments common to multiple speakers and utterances, any alignment observed less than 20% as often as the most frequent alignment for that canonical phone is discarded, which results in an average of three variants of each phone.[5]

4.2 Adapting the Recognizer

Now that we have probability distributions over observed phones, the HMMs trained on TIMIT are modified as follows to allow the observed variation. To allow, for instance, variation between `p` and `th`, the states for `th` from the original recognizer are inserted into the model for `p` as a separate path. The resulting phone model is shown in Figure 2. The transition probabilities into the first states

[2] Some of the spelling errors come from an elicitation task, so the distribution of target words is not representative of typical JWEFL productions, e.g., the corpus contains 102 different misspellings of `albatross`.

[3] SCOWL is available at http://wordlist.sourceforge.net.

[4] HTK is available at http://htk.eng.cam.ac.uk.

[5] There are 119 variants of 39 phones. The cutoff of 20% was chosen to allow a few variations for most phones. A small number of phones have no variants (e.g., `iy`, `w`) while a few have over nine variants (e.g., `ah`, `l`). It is not surprising that phones that are well-known to be difficult for Japanese speakers (cf. Minematsu et al., 2002) are the ones with the most variation.

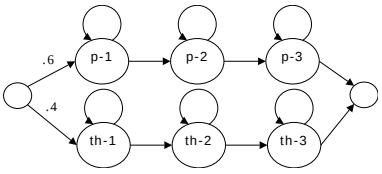

Figure 2: Adapted phone model for p accounting for variation between p and th

of the phones come from the probability distribution observed in the initial recognition step. The transition probabilities between the three states for each variant phone remain unchanged. All HMMs are adapted in this manner using the probability distributions from the initial recognition step.

The adapted HMMs are used to recognize the ERJ Corpus for a second time, this time in forced alignment with the canonical pronunciations. The state transitions indicate which variant of each phone was recognized and the correspondences between the canonical phones and recognized phones are used to generate a new probability distribution over observed phones for each canonical phone. These are used to find the most probable pronunciation variations for a native-speaker pronouncing dictionary.

4.3 Generating Pronunciations

The observed phone variation is used to generate multiple pronunciations for each pronunciation in the word list. The OpenFst Library[6] is used to find the most probable pronunciations in each case. First, FSTs are created for each phone using the probability distributions from the previous section. Next, an FST is created for the entire word by concatenating the FSTs for the pronunciation from CMUDICT. The pronunciations corresponding to the best n paths through the FST and the original canonical pronunciation become possible pronunciations in the extended pronouncing dictionary. The size 50 word list contains 54,001 words and when expanded to contain the top five variations of each pronunciation, there are 255,827 unique pronunciations.

5 Results

In order to evaluate the effect of pronunciation variation in Toutanova and Moore (2002)'s spelling correction approach, we compare the performance of the pronunciation model and the combined model with and without pronunciation variation.

We implemented the letter and pronunciation spelling correction models as described in section 2.2. The letter error model P_L and the phone error model P_{PH} are trained on the training set. The development set is used to tune the parameters introduced in previous sections.[7] In order to rank the words as candidate corrections for a misspelling r, $P_L(r|w)$ and $P_{PHL}(r|w)$ are calculated for each word in the word list using the algorithm described in Brill and Moore (2000). Finally, P_L and P_{PHL} are combined using S_{CMB} to rank each word.

5.1 Baseline

The open source spell checker *GNU Aspell*[8] is used to determine the baseline performance of a traditional spell checker using the same word list. An *Aspell* dictionary was created with the word list described in section 3. *Aspell*'s performance is shown in Table 1. The 1-Best performance is the percentage of test items for which the target word was the first candidate correction, 2-Best is the percentage for which the target was in the top two, etc.

5.2 Evaluation of Pronunciation Variation

The effect of introducing pronunciation variation using the method described in section 4 can be evaluated by examining the performance on the test set for P_{PHL} with and without the additional variations. The results in Table 1 show that the addition of pronunciation variations does indeed improve the performance of P_{PHL} across the board. The 1-Best, 3-Best, and 4-Best cases for P_{PHL} with variation show significant improvement ($p<0.05$) over P_{PHL} without variation.

5.3 Evaluation of the Combined Model

We evaluated the effect of including pronunciation variation in the combined model by comparing the performance of the combined model with and without pronunciation variation, see results in Table 1. Despite the improvements seen in P_{PHL} with pronunciation variation, there are no significant differences between the results for the combined model with and without variation. The combined model

[6] OpenFst is available at http://www.openfst.org/.

[7] The values are: $N = 3$ for the letter model, $N = 4$ for the phone model, $m = 80\%$, and $\lambda = 0.15$ in S_{CMB}.

[8] GNU Aspell is available at http://aspell.net.

Model	1-Best	2-Best	3-Best	4-Best	5-Best	6-Best
Aspell	44.1	54.0	64.1	68.3	70.0	72.5
Letter (L)	64.7	74.6	79.6	83.2	84.0	85.3
Pronunciation (PHL) without Pron. Var.	47.9	60.7	67.9	70.8	75.0	77.3
Pronunciation (PHL) with Pron. Var.	50.6	62.2	70.4	73.1	76.7	78.2
Combined (CMB) without Pron. Var.	64.9	75.2	78.6	81.1	82.6	83.2
Combined (CMB) with Pron. Var.	65.5	75.0	78.4	80.7	82.6	84.0

Table 1: Percentage of Correct Suggestions on the Atsuo-Henry Corpus Test Set for All Models

Rank	Aspell	L	PHL	CMB
1	enemy	enemy	**any**	enemy
2	envy	envy	Emmy	envy
3	energy	money	Ne	**any**
4	eye	emery	gunny	deny
5	teeny	deny	ebony	money
6	Ne	**any**	anything	emery
7	deny	nay	senna	nay
8	**any**	ivy	journey	ivy

Table 2: Misspelling `*eney`, Intended Word `any`

with variation is also not significantly different from the letter model P_L except for the drop in the 4-Best case.

To illustrate the performance of each model, the ranked lists in Table 2 give an example of the candidate corrections for the misspelling of `any` as `*eney`. *Aspell* preserves the initial letter of the misspelling and vowels in many of its candidates. P_L's top candidates also overlap a great deal in orthography, but there is more initial letter and vowel variation. As we would predict, P_{PHL} ranks `any` as the top correction, but some of the lower-ranked candidates for P_{PHL} differ greatly in length.

5.4 Summary of Results

The noisy channel spelling correction approach developed by Brill and Moore (2000) and Toutanova and Moore (2002) appears well-suited for writers of English as a foreign language. The letter and combined models outperform the traditional spell checker *Aspell* by a wide margin. Although including pronunciation variation does not improve the combined model, it leads to significant improvements in the pronunciation-based model P_{PHL}.

6 Conclusion

We have presented a method for modeling pronunciation variation from a phonetically untranscribed corpus of read non-native speech by adapting a monophone recognizer initially trained on native speech. This model allows a native pronouncing dictionary to be extended to include non-native pronunciation variations. We incorporated a pronouncing dictionary extended for Japanese writers of English into the spelling correction model developed by Toutanova and Moore (2002), which combines orthography-based and pronunciation-based models. Although the extended pronunciation dictionary does not lead to improvement in the combined model, it does leads to significant improvement in the pronunciation-based model.

Acknowledgments

I would like to thank Eric Fosler-Lussier, the Ohio State computational linguistics discussion group, and anonymous reviewers for their helpful feedback.

References

Brill, Eric and Robert C. Moore (2000). An Improved Error Model for Noisy Channel Spelling Correction. In *Proceedings of ACL 2000*.

CMUDICT (1998). CMU Pronouncing Dictionary version 0.6. http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

Fisher, Willam (1999). A statistical text-to-phone function using ngrams and rules. In *Proceedings of ICASSP 1999*.

Kukich, Karen (1992). Technique for automatically correcting words in text. *ACM Computing Surveys* 24(4).

Minematsu, N., Y. Tomiyama, K. Yoshimoto, K. Shimizu, S. Nakagawa, M. Dantsuji, and S. Makino (2002). English Speech Database Read by Japanese Learners for CALL System Development. In *Proceedings of LREC 2002*.

Mitton, Roger and Takeshi Okada (2007). The adaptation of an English spellchecker for Japanese writers. In *Symposium on Second Language Writing*.

Okada, Takeshi (2004). A Corpus Analysis of Spelling Errors Made by Japanese EFL Writers. *Yamagata English Studies* 9.

TIMIT (1991). TIMIT Acoustic-Phonetic Continuous Speech Corpus. NIST Speech Disc CD1-1.1.

Toutanova, Kristina and Robert Moore (2002). Pronunciation Modeling for Improved Spelling Correction. In *Proceedings of ACL 2002*.

Building a Semantic Lexicon of English Nouns via Bootstrapping

Ting Qian[1], **Benjamin Van Durme**[2] **and Lenhart Schubert**[2]
[1]Department of Brain and Cognitive Sciences
[2]Department of Computer Science
University of Rochester
Rochester, NY 14627 USA
ting.qian@rochester.edu, {vandurme, schubert}@cs.rochester.edu

Abstract

We describe the use of a weakly supervised bootstrapping algorithm in discovering contrasting semantic categories from a source lexicon with little training data. Our method primarily exploits the patterns in sentential contexts where different categories of words may appear. Experimental results are presented showing that such automatically categorized terms tend to agree with human judgements.

1 Introduction

There are important semantic distinctions between different types of English nouns. For example, some nouns typically refer to a concrete physical object, such as *book*, *tree*, etc. Others are used to represent the process or the result of an event (e.g. *birth*, *celebration*). Such information is useful in disambiguating syntactically similar phrases and sentences, so as to provide more accurate semantic interpretations. For instance, A MAN WITH HOBBIES and A MAN WITH APPLES share the same structure, but convey very different aspects about the *man* being referred to (i.e. activities vs possessions).

Compiling such a lexicon by hand, e.g., WordNet (Fellbaum, 1998), requires tremendous time and expertise. In addition, when new words appear, these will have to be analyzed and added manually. Furthermore, a single, global lexicon may contain erroneous categorizations when used within a specific domain/genre; we would like a "flexible" lexicon, adaptable to a given corpus. Also, in adapting semantic classifications of words to a particular genre or domain, we would like to be able to exploit continuing improvements in methods of extracting semantic occurrence patterns from text.

We present our initial efforts in discovering semantic classes incrementally under a weakly supervised bootstrapping process. The approach is able to selectively learn from its own discoveries, thereby minimizing the effort needed to provide seed examples as well as maintaining a reasonable accuracy rate. In what follows, we first focus on its application to an event-noun classification task, and then use a physical-object vs non-physical-object experiment as a showcase for the algorithm's generality.

2 Bootstrapping Algorithm

The bootstrapping algorithm discovers words with semantic properties similar to a small set of labelled seed examples. These examples can be manually selected from an existing lexicon. By simply changing the semantic property of the seed set, this algorithm can be applied to the task of discovering a variety of semantic classes.

Features Classification is performed using a perceptron-based model (Rosenblatt, 1958) that examines *features* of each word. We use two kinds of features in our model: morphological (affix and word length), and contextual. Suffixes, such as *-ion*, often reveal the semantic type that a noun belongs to (e.g., *destruction*, *explosion*). Other suffixes like *-er* typically suggest non-event nouns (e.g. *waiter*, *hanger*). The set of affixes can be modified to reflect meaningful distinctions in the task at hand. Regarding word length, longer words tend to have more

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 37–42,
Boulder, Colorado, June 2009.

syllables, and thus are more likely to contain affixes. For example, if a word ends with *-ment*, its number of letters must be ≥ 5. We defined a partition of words based on word length: shortest (fewer than 5 letters), short (5-7), medium (8-12), long (13-19), and longest (> 19).

Besides morphological features, we also make use of verbalized propositions resulting from the experiments of Van Durme et al. (2008) as contextual features. These outputs are in the form of world knowledge "factoids" abstracted from texts, based on logical forms from parsed sentences, produced by the KNEXT system (see Schubert (2002) for details). The followings are some sample factoids about the word *destruction*, extracted from the British National Corpus.

- A PERSON-OR-ORGANIZATION MAY UNDERGO A DESTRUCTION
- INDIVIDUAL -S MAY HAVE A DESTRUCTION
- PROPERTY MAY UNDERGO A DESTRUCTION

We take each verbalization (with the target word removed) as a contextual feature, such as `PROPERTY MAY UNDERGO A __`. Words from the same semantic category (e.g., event nouns) should have semantic and syntactic similarities on the sentential level. Thus their contextual features, which reflect the use of words both semantically and syntactically, should be similar. For instance, `PROPERTY MAY UNDERGO A PROTECTION` is another verbalization produced by KNEXT, suggesting the word *protection* may belong to the same category as *destruction*.

A few rough-and-ready heuristics are already employed by KNEXT to do the same task as we wish to automate here. A built-in classifier judges nominals to be event or non-event ones based on analysis of endings, plus a list of event nouns whose endings are unrevealing, and a list of non-event nouns whose endings tend to suggest they are event nouns. As a result, the factoids used as contextual features in our work already reflect the built-in classifier's attempt to distinguish event nouns from the rest. Thus, the use of these contextual features may bias the algorithm to perform seemingly well on event-noun classification. However, we will show that our algorithm works for classification of other semantic categories, for which KNEXT does not yet have discriminative procedures.

Iterative Training We use a bootstrapping procedure to iteratively train a perceptron-based linear classifier. A perceptron algorithm determines whether the active features of a test case are similar to those learned from given categories of examples. In an iterative training process, the classifier first learns from a small seed set, which contains examples of all categories (in binary classification, both positive and negative examples) manually selected to reflect human knowledge of semantic categories. The classifier then discovers new instances (and corresponding features) of each category. Based on activation values, these newly discovered instances are selectively admitted into the original training set, which increases the size of training examples for the next iteration.

The iterative training algorithm described above is adopted from Klementiev and Roth (2006). The advantage of bootstrapping is the ability to automatically learn from new discoveries, which saves both time and effort required to manually examine a source lexicon. However, if implemented exactly as described above, this process has two apparent disadvantages: New examples may be wrongly classified by the model; and it is difficult to evaluate the discriminative models produced in successive iterations, as there are no standard data against which to judge them (the new examples are by definition previously unexamined). We propose two measures to alleviate these problems. First, we admit into the training set only those instances whose activation values are higher than the mean activation of their corresponding categories in an iteration. This sets a variable threshold that is correlated with the performance of the model at each iteration. Second, we evaluate iterative results *post hoc*, using a *Bootstrapping Score*. This measures the efficacy of bootstrapping (i.e. the ratio of correct newly discovered instances to training examples) and precision (i.e. the proportion of correct discoveries among all those returned by the algorithm). We compute this score to decide which iteration has yielded the optimal discriminative model.

3 Building an Event-noun Lexicon

We applied the bootstrapping algorithm to the task of discovering event nouns from a source lexicon.

Event nouns are words that typically describe the occurrence, the process, or the result of an event. We first explore the effectiveness of this algorithm, and then describe a method of extracting the optimal model. Top-ranked features in the optimal model are used to find subcategories of event nouns.

Experimental Setup The WordNet noun-list is chosen as the source lexicon (Fellbaum, 1998), which consists of 21,512 nouns. The purpose of this task is to explore the separability of event nouns from this collection.

typical suffixes: *appeasement, arrival, renewal, construction, robbery, departure, happening*
irregular cases: *birth, collapse, crash, death, decline, demise, loss, murder*

Table 1: Examples of event-nouns in initial training set.

We manually selected 15 event nouns and 215 non-event nouns for the seed set. Event-noun examples are representative of subcategories within the semantic class, as well as their commonly seen morphological structures (Table 1). Non-event examples are primarily exceptions to morphological regularities (to prevent the algorithm from overly relying on affix features), such as, *anything*, *ambition*, *diagonal*. The subset of all contextual and morphological features represented by both event and non-event examples are used to bootstrap the training process.

Event Noun Discovery Reducing the number of working features is often an effective strategy in training a perceptron. We experimented with two cut-off thresholds for features: in Trial 1, features must appear at least 10 times (55,354 remaining); in Trial 2, features must appear at least 15 times (35,457 remaining).

We set the training process to run for 20 iterations in both trials. Classification results of each iteration were collected. We expect the algorithm to discover few event nouns during early iterations. But with new instances found in each subsequent iteration, it ought to utilize newly seen features and discover more. Figure 1 confirms our intuition.

The number of classified event-noun instances increased sharply at the 15th iteration in Trial 1 and the 11th iteration in Trial 2, which may suggest overfitting of the training examples used in those iterations. If so, this should also correlate with an increase of error rate in the classification results (error rate defined as the percentage of non-event nouns identified as event nouns in all discovered event nouns). We manually marked all misclassified event noun instances for the first 10 iterations in both trials. The error rate in Trial 2 is expected to significantly increase at the 10th iteration, while Trial 1 should exhibit little increase in error rate within this interval. This expectation is confirmed in Figure 2.

Extracting the Optimal Model We further pursued the task of finding the iteration that has yielded the best model. Optimality is judged from two aspects: 1) the number of correctly identified event nouns should be significantly larger than the size of seed examples; and 2) the accuracy of classification results should be relatively high so that it takes little effort to clean up the result. Once the optimal model is determined, we analyze its most heavily weighted features and try to derive finer categories from them. Furthermore, the optimal model could be used to discover new instances from other source lexicons in the future.

We define a measure called the Bootstrapping Score (BS), serving a similar purpose as an F-score. BS is computed as in Formula (1).

$$BS = \frac{2 * BR * Precision}{BR + Precision}\,. \qquad (1)$$

Here the Bootstrapping Rate (BR) is computed as:

$$BR = \frac{|NEW|}{|NEW| + |SEED|}\,, \qquad (2)$$

where $|NEW|$ is the number of correctly identified new instances (seed examples excluded), and $|SEED|$ is the size of seed examples. The rate of bootstrapping reveals how large the effect of the bootstrapping process is. Note that BR is different from the classic measure *recall*, for which the total number of relevent documents (i.e. true event nouns in English) must be known *a priori* – again, this knowledge is what we are discovering. The score is a *post hoc* solution; both BR and precision are computed for analysis after the algorithm has finished. Combining Formulas (1) and (2), a higher Bootstrapping Score means better model quality.

Bootstrapping scores of models in the first ten iterations are plotted in Figure 3. Model quality in

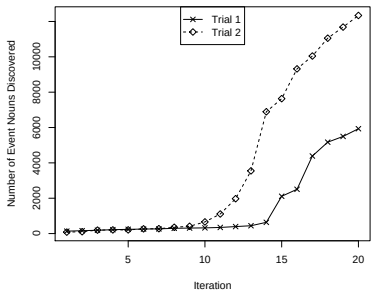

Figure 1: Classification rate

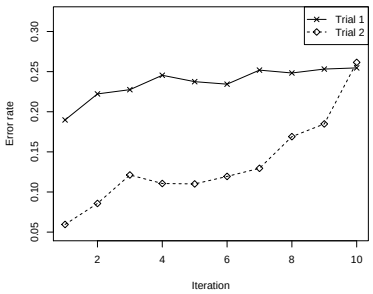

Figure 2: Error rate

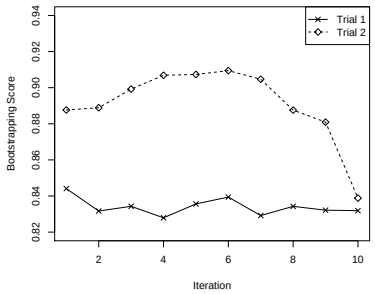

Figure 3: Bootstrapping score

	1	...	**6**	...	**10**
incorrect	5	...	32	...	176
correct	79	...	236	...	497
error rate	5.9%	...	11.9%	...	26.2%
score	87.0%	...	90.8%	...	83.8%

Table 2: From iterations 1 to 10, comparison between instance counts, error rates, and bootstrapping scores as the measure of model quality.

Trial 2 is better than in Trial 1 on average. In addition, within Trial 2, Iteration 6 yielded the best discriminative model with a bootstrapping score of 90.8%. Compared to instance counts and error rate measures as shown in Table 2, this bootstrapping score provides a balanced measure of model quality. The model at the 6th iteration (hereafter, Model 6) can be considered the optimal model generated during the bootstrapping training process.

Top-ranked Features in the Optimal Model In order to understand why Model 6 is optimal, we extracted its top 15 features that activate the event-noun target in Model 6, as listed in Table 3. Interestingly, top-ranked features are all contextual ones. In fact, in later models where the ranks of morphological features are boosted, the algorithm performed worse as a result of relying too much on those context-insensitive features.

Collectively, top-ranked features define the contextual patterns of event nouns. We are interested in finding semantic subcategories within the set of event nouns (497 nouns, Trial 2) by exploiting these features individually. For instance, some events typically happen to people only (e.g. *birth*, *betrayal*), while others usually happen to inanimate objects (e.g. *destruction*, *removal*). Human actions can also be distinguished by the number of participants, such as group activities (e.g. *election*) or individual activities (e.g. *death*). It is thus worth distinguishing nouns that describe different sorts of events.

Manual Classification We extracted the top 100 contextual features from Model 6 and grouped them into *feature classes*. A feature class consists of contextual features sharing similar meanings. For instance, `A COUNTRY MAY UNDERGO __` and `A STATE MAY UNDERGO __` both belong to the class *social activity*. For each feature class, we enumerate all words that correspond to its feature instances. Examples are shown in Table 4.

Not all events can be unambiguously classified into one of the subcategories. However, this is also not necessary because these categories overlap with one another. For example, *death* describes an event that tends to occur both individually and briefly. In addition to the six categories listed here, new categories can be added by creating more feature classes.

Automatic Clustering Representing each noun as a frequency vector over the top 100 most discriminating contextual features, we employed k-means clustering and compared the results to our manually crafted subcategories.

Through trial-and-error, we set k to 12, with the smallest resulting cluster containing 2 nouns (*interpretation* and *perception*), while the biggest resulting cluster contained 320 event nouns (that seemed to share no apparent semantic properties). Other clusters varied from 5 to 50 words in size, with examples shown in Table 5.

The advantage of automatic clustering is that the results may reflect an English speaker's impression of word similarity gained through language use. Un-

a person-or-organization may undergo a _	a state may undergo a _	a _ can be attempted
a country may undergo a _	a child may have a _	a _ can be for a country
a company may undergo a _	a project may undergo a _	authority may undergo a _
an explanation can be for a _	an empire may undergo a _	a war may undergo a _
days may have a _	a _ can be abrupt	a _ can be rapid

Table 3: Top 15 features that promote activation of the event-noun target, ranked from most weighted to least.

human events: adoption, arrival, birth, betrayal, death, development, disappearance, emancipation, funeral ...
events of inanimate objects: collapse, construction, definition, destruction, identification, inception, movement, recreation, removal ...
individual activities: birth, death, execution, funeral, promotion ...
social activities: abolition, evolution, federation, fragmentation, invasion ...
lasting events: campaign, development, growth, trial ...
brief events: awakening, collapse, death, mention, onset, resignation, thunderstorm ...

Table 4: Six subcategories of event nouns.

Cluster 1 (17): cancellation, cessation, closure, crackle, crash, demise, disappearance, dismissal, dissolution, division, introduction, onset, passing, resignation, reversal, termination, transformation
Cluster 2 (32): alienation, backing, betrayal, contemplation, election, execution, funeral, hallucination, imitation, juxtaposition, killing, mention, moulding, perfection, prosecution, recognition, refusal, removal, resurrection, semblance, inspection, occupation, promotion, trial ...
Cluster 3 (7): development, achievement, arrival, birth, death, loss, survival

Table 5: Examples resulting from automatic clustering.

fortunately, the discovered clusters do not typically come with the same obvious semantic properties as were defined in manual classification. In the example given above, neither of Cluster 1 and Cluster 3 seems to have a centralized semantic theme. But Cluster 2 seems to be mostly about human activities.

Comparison with WordNet To compare our results with WordNet resources, we enumerated all children of the gloss "*something that happens at a given place and time*", giving 7655 terms (phrases excluded). This gave a broader range of event nouns, such as proper nouns and procedures (e.g. *9/11*, *CT*, *MRI*), onomatopoeias (e.g. *mew*, *moo*), and words whose event reading is only secondary (e.g. *picture*, *politics*, *teamwork*). These types of words tend to have very different contextual features from what our algorithm had discovered.

While our method may be limited by the choice of seed examples, we were able to discover event nouns not classified under this set by WordNet, suggesting that the discovery mechanism itself is a robust one. Among them were low-frequency nouns (e.g. *crescendo*, *demise*, names of processes (e.g. *absorption*, *evolution*), and particular cases like *thunderstorm*.

4 Extension to Other Semantic Categories

To verify that our bootstrapping algorithm was not simply relying on KNEXT's own event classification heuristics, we set the algorithm to learn the distinction between physical and non-physical objects/entities.

(Non-)Physical-object Nouns 15 physical-object/entity nouns (e.g. *knife*, *ring*, *anthropologist*) and 34 non-physical ones (e.g. *happiness*, *knowledge*) were given to the model as the initial training set. At the 9th iteration, the number of discovered physical objects (which form the minority group between the two) approaches 2,200 and levels off. We randomly sampled five 20-word groups (a subset of these words are listed in Table 6) from this entire set of discovered physical objects, and computed an average error rate of 4%. Prominent features of the model at the 9th iteration are shown in Table 7.

5 Related Work

The method of using distributional patterns in a large text corpus to find semantically related En-

heifer, sheriff, collector, hippie, accountant, cape, scab, pebble, box, dick, calculator, sago, brow, ship, *?john*, superstar, border, rabbit, poker, garter, grinder, millionaire, ash, herdsman, *?cwm*, pug, bra, fulmar, **campaign*, stallion, deserter, boot, tear, elbow, cavalry, novel, cardigan, nutcase, *?bulge*, businessman, cop, fig, musician, spire, butcher, dog, elk, ...

Table 6: Physical-object nouns randomly sampled from results; words with an asterisk are misclassified, ones with a question mark are doubtful.

a male-individual can be a _	a _ can be small
a person can be a _	a _ can be large
a _ can be young	a _ can be german
-S**morphological feature*	a _ can be british
a _ can be old	a _ can be good

Table 7: Top-10 features that promote activation of the physical-object target in the model.

glish nouns first appeared in Hindle (1990). Roark and Charniak (1998) constructed a semantic lexicon using co-occurrence statistics of nouns within noun phrases. More recently, Liakata and Pulman (2008) induced a hierarchy over nominals using as features knowledge fragments similar to the sort given by KNEXT. Our work might be viewed as aiming for the same goal (a lexico-semantic based partitioning over nominals, tied to corpus-based knowledge), but allowing for an *a priori* bias regarding preferred structure.

The idea of bootstrapping lexical semantic properties goes back at least to Hearst (1998), where the idea is suggested of using seed examples of a relation to discover lexico-syntactic extraction patterns and then using these to discover further examples of the desired relation. The Basilisk system developed by Thelen and Riloff (2002) almost paralleled our effort. However, negative features – features that would prevent a word from being classified into a semantic category – were not considered in their model. In addition, in scoring candidate words, their algorithm only looked at the average relevance of syntactic patterns. Our perceptron-based algorithm examines the combinatorial effect of those patterns, which has yielded results suggesting improved accuracy and bootstrapping efficacy.

Similar to our experiments here using k-means, Lin and Pantel (2001) gave a clustering algorithm for iteratively building semantic classes, using as features argument positions within fragments from a syntactic dependency parser.

6 Conclusion

We have presented a bootstrapping approach for creating semantically tagged lexicons. The method can effectively classify nouns with contrasting semantic properties, even when the initial training set is a very small. Further classification is possible with both manual and automatic methods by utilizing individual contextual features in the optimal model.

Acknowledgments

This work was supported by NSF grants IIS-0328849 and IIS-0535105.

References

BNC Consortium. 2001. The British National Corpus, version 2 (BNC World). Distributed by Oxford University Computing Services.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Marti A. Hearst. 1998. Automated discovery of WordNet relations. In (Fellbaum, 1998), pages 131–153.

Donald Hindle. 1990. Noun classification from predicate-argument structures. In *ACL*.

Alexandre Klementiev and Dan Roth. 2006. Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In *ACL*.

Maria Liakata and Stephen Pulman. 2008. Automatic Fine-Grained Semantic Classification for Domain Adaption. In *Proceedings of Semantics in Text Processing (STEP)*.

Dekang Lin and Patrick Pantel. 2001. Induction of semantic classes from natural language text. In *KDD*.

Brian Roark and Eugene Charniak. 1998. Noun-phrase co-occurrence statistics for semi-automatic semantic lexicon construction. In *ACL*, pages 1110–1116.

Frank Rosenblatt. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408.

Lenhart K. Schubert. 2002. Can we derive general world knowledge from text? In *HLT*.

Michael Thelen and Ellen Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *EMNLP*.

Benjamin Van Durme, Ting Qian, and Lenhart Schubert. 2008. Class-driven Attribute Extraction. In *COLING*.

Multiple Word Alignment with Profile Hidden Markov Models

Aditya Bhargava and Grzegorz Kondrak
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada, T6G 2E8
{abhargava,kondrak}@cs.ualberta.ca

Abstract

Profile hidden Markov models (Profile HMMs) are specific types of hidden Markov models used in biological sequence analysis. We propose the use of Profile HMMs for word-related tasks. We test their applicability to the tasks of multiple cognate alignment and cognate set matching, and find that they work well in general for both tasks. On the latter task, the Profile HMM method outperforms average and minimum edit distance. Given the success for these two tasks, we further discuss the potential applications of Profile HMMs to any task where consideration of a set of words is necessary.

1 Introduction

In linguistics, it is often necessary to align words or phonetic sequences. Covington (1996) uses alignments of cognate pairs for the historical linguistics task of comparative reconstruction and Nerbonne and Heeringa (1997) use alignments to compute relative distances between words from various Dutch dialects. Algorithms for aligning pairs of words have been proposed by Covington (1996) and Kondrak (2000). However, it is often necessary to align multiple words. Covington (1998) proposed a method to align multiple words based on a hand-crafted scale of similarity between various classes of phonemes, again for the purpose of comparative reconstruction of languages.

Profile hidden Markov models (Profile HMMs) are specific types of hidden Markov models used in biological sequence analysis, where they have yielded success for the matching of given sequences to sequence families as well as to multiple sequence alignment (Durbin et al., 1998). In this paper, we show that Profile HMMs can be adapted to the task of aligning multiple words. We apply them to sets of multilingual cognates and show that they produce good alignments. We also use them for the related task of matching words to established cognate sets, useful for a situation where it is not immediately obvious to which cognate set a word should be matched. The accuracy on the latter task exceeds the accuracy of a method based on edit distance.

Profile HMMs could also potentially be used for the computation of word similarity when a word must be compared not to another word but to another set of words, taking into account properties of all constituent words. The use of Profile HMMs for multiple sequence alignment also presents applications to the acquisition of mapping dictionaries (Barzilay and Lee, 2002) and sentence-level paraphrasing (Barzilay and Lee, 2003).

This paper is organized as follows: we first describe the uses of Profile HMMs in computational biology, their structure, and then discuss their applications to word-related tasks. We then discuss our data set and describe the tasks that we test and their experimental setups and results. We conclude with a summary of the results and a brief discussion of potential future work.

2 Profile hidden Markov models

In computational biology, it is often necessary to deal with multiple sequences, including DNA and protein sequences. For such biological sequence analysis, Profile HMMs are applied to the common tasks of simultaneously aligning multiple related sequences to each other, aligning a new sequence to

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 43–48,
Boulder, Colorado, June 2009.

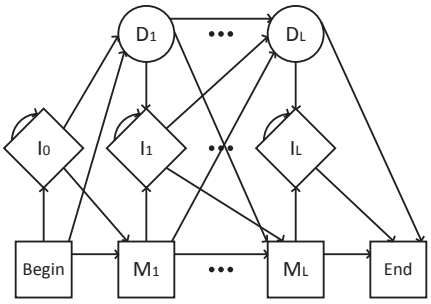

Figure 1: A prototypical Profile HMM of length *L*. M_i is the ith match state, I_i is the ith insert state, and D_i is the ith delete state. Delete states are silent and are used to indicate gaps in a sequence.

an already-aligned family of sequences, and evaluating a new sequence for membership in a family of sequences.

Profile HMMs consist of several types of states: match states, insert states, delete states, as well as a begin and end state. For each position in a Profile HMM, there is one match state, one insert state, and one delete state. A Profile HMM can thus be visualized as a series of columns, where each column represents a position in the sequence (see Figure 1). Any arbitrary sequence can then be represented as a traversal of states from column to column.

Match states form the core of the model; each match state is represented by a set of emission probabilities for each symbol in the output alphabet. These probabilities indicate the distribution of values for a given position in a sequence. Each match state can probabilistically transition to the next (i.e. next-column) match and delete states as well as the current (i.e. current-column) insert state.

Insert states represent possible values that can be inserted at a given position in a sequence (before a match emission or deletion). They are represented in the same manner as match states, with each output symbol having an associated probability. Insert states are used to account for symbols that have been inserted to a given position that might not otherwise have occurred "naturally" via a match state. Insert states can probabilistically transition to the next match and delete states as well as the current insert state (i.e. itself). Allowing insert states to transition to themselves enables the consideration of multiple-symbol inserts.

```
MMIIIM
AG...C
A-AG.C
AG.AA-
--AAAC
AG...C
```

Figure 2: A small DNA multiple alignment from (Durbin et al., 1998, p. 123).

Similarly, delete states represent symbols that have been removed from a given position. For a sequence to use a delete state for a given position indicates that a given character position in the model has no corresponding characters in the given sequence. Hence, delete states are by nature silent and thus have no emission probabilities for the output symbols. This is an important distinction from match states and insert states. Each delete state can probabilistically transition to the next match and delete states as well as the current insert state.

Figure 2 shows a small example of a set of DNA sequences. The match columns and insert columns are marked with the letters M and I respectively in the first line. Where a word has a character in a match column, it is a match state emission; when there is instead a gap, it is a delete state occurrence. Any characters in insert columns are insert state emissions, and gaps in insert columns represent simply that the particular insert state was not used for the sequence in question.

Durbin et al. (1998) describe the uses of Profile HMMs for tasks in biological sequence analysis. Firstly, a Profile HMM must be constructed. If a Profile HMM is to be constructed from a set of aligned sequences, it is necessary to designate certain columns as match columns and others as insert column. The simple heuristic that we adopt is to label those columns match states for which half or more of the sequences have a symbol present (rather than a gap). Other columns are labelled insert states. Then the probability a_{kl} of state k transitioning to state l can be estimated by counting the number of times A_{kl} that the transition is used in the alignment:

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}}$$

Similarly, the probability $e_k(a)$ of state k emitting symbol a is estimated by counting the number of

times $E_k(a)$ that the emission is used in the alignment:

$$e_k(a) = \frac{E_k(a)}{\sum_{a'} E_k(a')}$$

There is the danger that some probabilities may be set to zero, so it is essential to add pseudocounts. The pseudocount methods that we explore are described in section 3.

If a Profile HMM is to be constructed from a set of unaligned sequences, an initial model is generated after which it can be trained to the sequences using the Baum-Welch algorithm. The length of the model must be chosen, and is usually set to the average length of the unaligned sequences. To generate the initial model, which amounts to setting the transition and emission probabilities to some initial values, the probabilities are sampled from Dirichlet distributions.

Once a Profile HMM has been constructed, it can be used to evaluate a given sequence for membership in the family. This is done via a straightforward application of the forward algorithm (to get the full probability of the given sequence) or the Viterbi algorithm (to get the alignment of the sequence to the family). For the alignment of multiple unaligned sequences, a Profile HMM is constructed and trained as described above and then each sequence can be aligned using the Viterbi algorithm.

It should also be noted that Profile HMMs are generalizations of Pair HMMs, which have been used for cognate identification and word similarity (Mackay and Kondrak, 2005) between pairs of words. Unlike Pair HMMs, Profile HMMs are position-specific; this is what allows their application to multiple sequences but also means that each Profile HMM must be trained to a given set of sequences, whereas Pair HMMs can be trained over a very large data set of pairs of words.

3 Adapting Profile HMMs to words

Using Profile HMMs for biological sequences involves defining an alphabet and working with related sequences consisting of symbols from that alphabet. One could perform tasks with cognates sets in a similar manner; cognates are, after all, related words, and words are nothing more than sequences of symbols from an alphabet. Thus Profile HMMs present potential applications to similar tasks for cognate sets. We apply Profile HMMs to the multiple alignment of cognate sets, which is done in the same manner as multiple sequence alignment for biological sequences described above. We also test Profile HMMs for determining the correct cognate set to which a word belongs when given a variety of cognate sets for the same meaning; this is done in a similar manner to the sequence membership evaluation task described above.

Although there are a number of Profile HMM packages available (e.g. HMMER), we decided to develop an implementation from scratch in order to achieve greater control over various adjustable parameters.[1] We investigated the following parameters:

Favouring match states When constructing a Profile HMM from unaligned sequences, the choice of initial model probabilities can have a significant effect on results. It may be sensible to favour match states compared to other states when constructing the initial model; since the transition probabilities are sampled from a Dirichlet distribution, the option of favouring match states assigns the largest returned probability to the transition to a match state.

Pseudocount method We implemented three pseudocount methods from (Durbin et al., 1998). In the following equations, $e_j(a)$ is the probability of state j emitting character a. c_{ja} represents the observed counts of state j emitting symbol a. A is the weight given to the pseudocounts.

Constant value A constant value AC is added to each count. This is a generalization of Laplace's rule, where $C = \frac{1}{A}$.

$$e_j(a) = \frac{c_{ja} + AC}{\sum_{a'} c_{ja'} + A}$$

Background frequency Pseudocounts are added in proportion to the background frequency q_a, which is the frequency of occurrence of character a.

$$e_j(a) = \frac{c_{ja} + Aq_a}{\sum_{a'} c_{ja'} + A}$$

[1] Our implementation is available online at `http://www.cs.ualberta.ca/~ab31/profilehmm`.

Substitution matrix (Durbin et al., 1998) Given a matrix $s(a, b)$ that gives the log-odds similarity of characters a and b, we can determine the conditional probability of a character b given character a:

$$P(b|a) = q_b e^{s(a,b)}$$

Then we define f_{ja} to be the probability derived from the counts:

$$f_{ja} = \frac{c_{ja}}{\sum_{a'} c_{ja'}}$$

Then the pseudocount values are set to:

$$\alpha_{ja} = A \sum_b f_{jb} P(a|b)$$

Finally, the pseudocount values are added to the real counts as above:

$$e_j(a) = \frac{c_{ja} + \alpha_{ja}}{\sum_{a'} c_{ja'} + \alpha_{ja'}}$$

Pseudocount weight The weight that the pseudocounts are given (A in the above equations).

Smoothing during Baum-Welch The problem has many local optima and it is therefore easy for the Baum-Welch algorithm to get stuck around one of these. In order to avoid local optima, we tested the option of adding pseudocounts during Baum-Welch (i.e. between iterations) rather than after it. This serves as a form of noise injection, effectively bumping Baum-Welch away from local optima.

4 Data for experiments

Our data come from the Comparative Indoeuropean Data Corpus (Dyen et al., 1992). The data consist of words in 95 languages in the Indoeuropean family organized into word lists corresponding to one of 200 meanings. Each word is represented in the English alphabet. Figure 3 shows a sample from the original corpus data. We manually converted the data into disjoint sets of cognate words, where each cognate set contains only one word from each language. We also removed words that were not cognate with any other words.

On average, there were 4.37 words per cognate set. The smallest cognate set had two words (since we excluded those words that were not cognate with any other words), and the largest had 84 words. There were on average 10.92 cognate sets in a meaning. The lowest number of cognate sets in a meaning was 1, and the largest number was 22.

```
a 026 DAY
...
b                       003
  026 53 Bulgarian      DEN
  026 47 Czech E        DENY
  026 45 Czech          DEN
  026 43 Lusatian L     ZEN
  026 44 Lusatian U     DZEN
  026 50 Polish         DZIEN
  026 51 Russian        DEN
  026 54 Serbocroatian  DAN
  026 42 Slovenian      DAN
  026 41 Latvian        DIENA
  026 05 Breton List    DEIZ, DE(Z)
  026 04 Welsh C        DYDD
  026 20 Spanish        DIA
  026 17 Sardinian N    DIE
  026 11 Ladin          DI
  026 08 Rumanian List  ZI
  026 09 Vlach          ZUE
  026 15 French Creole C ZU
  026 13 French         JOUR
  026 14 Walloon        DJOU
  026 10 Italian        GIORNO
...
```

Figure 3: An excerpt from the original corpus data. The first two numbers denote the meaning and the language, respectively.

5 Multiple cognate alignment

Similar to their use for multiple sequence alignment of sequences in a family, we test Profile HMMs for the task of aligning cognates. As described above, an initial model is generated. We use the aforementioned heuristic of setting the initial model length to the average length of the sequences. The transition probabilities are sampled from a uniform-parameter Dirichlet distribution, with each parameter having a value of 5.0. The insert-state emission probabilities are set to the background frequencies and the match-state emission probabilities are sampled from a Dirichlet distribution with parameters set in proportion to the background frequency. The model is

MIIMIIMI	MIIMIIMI
D--E--N-	D--E--NY
Z--E--N-	DZ-E--N-
DZIE--N-	D--A--N-
DI-E--NA	D--E--IZ
D--I--A-	D--Y--DD
D--I--E-	Z-----U-
Z--U--E-	Z-----I-
J--O--UR	D-----I-
DJ-O--U-	G--IORNO

Figure 4: The alignment generated via the Profile HMM method for some cognates. These were aligned together, but we show them in two columns to preserve space.

trained to the cognate set via the Baum-Welch algorithm, and then each word in the set is aligned to the model using the Viterbi algorithm. The words are added to the training via a summation; therefore, the order in which the words are considered has no effect, in contrast to iterative pairwise methods.

The setting of the parameter values is discussed in section 6.

5.1 Results

To evaluate Profile HMMs for multiple cognate alignment, we analyzed the alignments generated for a number of cognate sets. We found that increasing the pseudocount weight to 100 improved the quality of the alignments by effectively biasing the model towards similar characters according to the substitution matrix.

Figure 4 shows the Profile HMM alignment for a cognate set of words with the meaning "day." As with Figure 2, the alignment's first line is a guide label used to indicate which columns are match columns and which are insert columns; note that consecutive insert columns represent the same insert state and so are not aligned by the Profile HMM. While there were some duplicate words (i.e. words that had identical English orthographic representations but came from different languages), we do not show them here for brevity.

In this example, we see that the Profile HMM manages to identify those columns that are more highly conserved as match states. The ability to identify characters that are similar and align them correctly can be attributed to the provided substitution matrix.

Note that the characters in the insert columns should not be treated as aligned even though they represent emissions from the same insert state (this highlights the difference between match and insert states). For example, Y, A, Z, D, R, and O are all placed in a single insert column even though they cannot be traced to a single phoneme in a protoform of the cognate set. Particularly infrequent characters are more likely to be put together than separated even if they are phonetically dissimilar.

There is some difficulty, also evident from other alignments we generated, in isolating phonemes represented by pairs of characters (digraphs) as singular entities. In the given example, this means that the *dz* in *dzien* was modelled as a match state and then an insert state. This is, however, an inherent difficulty in using data represented only with the English alphabet, which could potentially be addressed if the data were instead represented in a standard phonetic notation such as IPA.

6 Cognate set matching

Evaluating alignments in a principled way is difficult because of the lack of a gold standard. To adjust for this, we also evaluate Profile HMMs for the task of matching a word to the correct cognate set from a list of cognate sets with the same meaning as the given word, similar to the evaluation of a biological sequence for membership in a family. This is realized by removing one word at a time from each word list and then using the resulting cognate sets within the meaning as possible targets. A model is generated from each possible target and a log-odds score is computed for the word using the forward algorithm. The scores are then sorted and the highest score is taken to be the cognate set to which the given word belongs. The accuracy is then the fraction of times the correct cognate set is identified.

To determine the best parameter values, we used a development set of 10 meanings (roughly 5% of the data). For the substitution matrix pseudocount method, we used a log-odds similarity matrix derived from Pair HMM training (Mackay and Kondrak, 2005). The best results were achieved with favouring of match states enabled, substitution-matrix-based pseudocount, pseudocount weight of 0.5, and pseudocounts added during Baum-Welch.

6.1 Results

We employed two baselines to generate scores between a given word and cognate set. The first baseline uses the average edit distance of the test word and the words in the given cognate set as the score of the word against the set. The second baseline is similar but uses the minimum edit distance between the test word and any word in the given cognate set as the score of the word against the entire set. For example, in the example set given in Figure 4, the average edit distance between *zen* and all other words in the set is 2.58 (including the hidden duplicate words) and the minimum edit distance is 1. All other candidate sets are similarly scored and the one with the lowest score is considered to be the correct cluster with ties broken randomly.

With the parameter settings described in the previous section, the Profile HMM method correctly identifies the corresponding cognate set with an accuracy of 93.2%, a substantial improvement over the average edit distance baseline, which obtains an accuracy of 77.0%.

Although the minimum edit distance baseline also yields an impressive accuracy of 91.0%, its score is based on a single word in the candidate set, and so would not be appropriate for cases where consideration of the entire set is necessary. Furthermore, the baseline benefits from the frequent presence of duplicate words in the cognate sets. Profile HMMs are more robust, thanks to the presence of identical or similar characters in corresponding positions.

7 Conclusions

Profile HMMs present an approach for working with sets of words. We tested their use for two cognate-related tasks. The method produced good-quality multiple cognate alignments, and we believe that they could be further improved with phonetically transcribed data. For the task of matching words to correct cognate sets, we achieved an improvement over the average edit distance and minimum edit distance baselines.

Since Profile HMM training is highly sensitive to the choice of initial model, we would like to explore more informed methods of constructing the initial model. Similarly, for building models from unaligned sequences, the addition of domain knowledge would likely prove beneficial. We also plan to investigate better pseudocount methods, as well as the possibility of using n-grams as output symbols.

By simultaneously considering an entire set of related words, Profile HMMs provide a distinct advantage over iterative pairwise methods. The success on our tasks of multiple alignment and cognate set matching suggests applicability to similar tasks involving words, such as named entity recognition across potentially multi-lingual corpora.

Acknowledgements

We thank Qing Dou for organizing the cognate sets from the original data. We are also grateful to the anonymous reviewers for their valuable comments. This research was funded in part by the Natural Sciences and Engineering Research Council of Canada.

References

Regina Barzilay and Lillian Lee. 2002. Bootstrapping lexical choice via multiple-sequence alignment. In *Proc. of EMNLP*, pages 164–171.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proc. of NAACL-HLT*, pages 16–23.

Michael A. Covington. 1996. An algorithm to align words for historical comparison. *Computational Linguistics*, 22(4):481–496.

Michael A. Covington. 1998. Alignment of multiple languages for historical comparison. In *Proc. of COLING-ACL*, pages 275–279.

Richard Durbin, Sean R. Eddy, Anders Krogh, and Graeme Mitchison. 1998. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge University Press.

Isidore Dyen, Joseph B. Kruskal, and Paul Black. 1992. An Indoeuropean classification: A lexicostatistical experiment. *Transactions of the American Philosophical Society*, 82(5).

Grzegorz Kondrak. 2000. A new algorithm for the alignment of phonetic sequences. In *Proc. of NAACL*, pages 288–295.

Wesley Mackay and Grzegorz Kondrak. 2005. Computing word similarity and identifying cognates with pair hidden Markov models. In *Proc. of CoNLL*, pages 40–47.

John Nerbonne and Wilbert Heeringa. 1997. Measuring dialect distance phonetically. In *Proc. of the Third Meeting of ACL SIGPHON*.

Using emotion to gain rapport in a spoken dialog system

Jaime C. Acosta
Department of Computer Science
University of Texas at El Paso
El Paso, TX 79968, USA
jcacosta@miners.utep.edu

Abstract

This paper describes research on automatically building rapport. This is done by adapting responses in a spoken dialog system to users' emotions as inferred from nonverbal voice properties. Emotions and their acoustic correlates will be extracted from a persuasive dialog corpus and will be used to implement an emotionally intelligent dialog system; one that can recognize emotion, choose an optimal strategy for gaining rapport, and render a response that contains appropriate emotion, both lexically and auditory. In order to determine the value of emotion modeling for gaining rapport in a spoken dialog system, the final implementation will be evaluated using different configurations through a user study.

1 Introduction

As information sources become richer and technology advances, the use of computers to deliver information is increasing. In particular, interactive voice technology for information delivery is becoming more common due to improvements in technologies such as automatic speech recognition, and speech synthesis.

Several problems exist in these voice technologies including speech recognition accuracy and lack of common sense and basic knowledge. Among these problems is the inability to achieve rapport.

Gratch *et al.* (2007) defines rapport as *a feeling of connectedness that seems to arise from rapid and contingent positive feedback between partners and is often associated with socio-emotional processes*. In the field of neuro-linguistics, O'Connel and Seymour (1990) stated that matching or complimenting voice features such as volume, speed, and intonation, is important to gain rapport. Shepard et al.'s Communication Accommodation Theory (2001) states that humans use prosody and backchannels in order to adjust social distance with an interlocutor. These features of voice can also be associated with emotions.

Previous work has shown that automated systems can gain rapport by reacting to user gestural nonverbal behavior (Chartrand and Bargh, 1999; Gratch et al., 2007; Cassell and Bickmore, 2003). In contrast, this research looks at how rapport can be gained through voice-only interaction.

Preliminary analysis of human-human dialog provides evidence that shifts in pitch, associated with emotion by two judges, are used by an interlocutor for persuasion. Figure 1 shows the pitch of a sound snippet from the corpus and how it differs from neutral, computer synthesized voice (produced using MaryTTS). This illustrates the more general fact that when humans speak to each other, we display a variety of nonverbal behaviors in voice, especially when trying to build rapport. The main hypothesis of this research is that a spoken dialog system with emotional intelligence will be effective for gaining rapport with human users.

The rest of this paper is structured as follows: first, related work is reviewed and current limitations for building automated rapport are described. Afterwards, the hypotheses and expected contributions of this work are described along with the research approach. Lastly, broader significance of this work is discussed.

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 49–54,
Boulder, Colorado, June 2009.

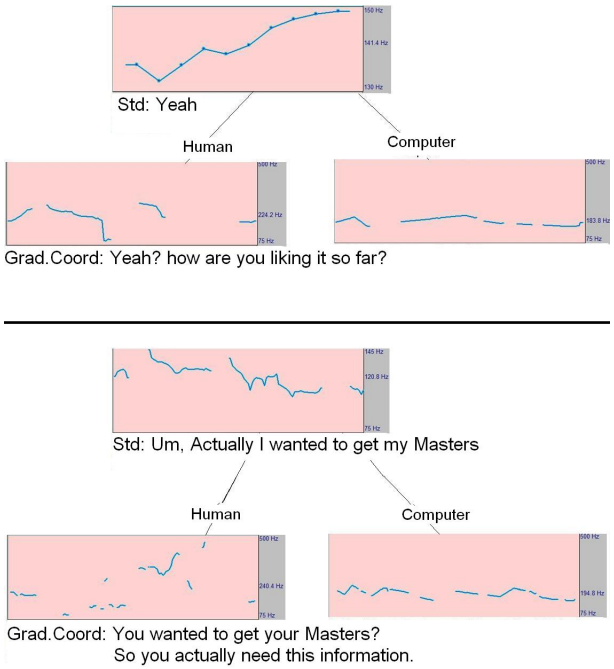

Figure 1: Pitch levels of a conversation taken from the persuasive dialog corpus includes a student (Std) and a graduate coordinator (Grad.Coord). Pitch was analyzed using the Praat software. It can be seen that the student displays rich prosody in voice (tree parents) and that the human response (left branch) contains more varied prosody than the computer synthesized voice (right branch).

2 Related Work

Communication Accommodation Theory states that people use nonverbal feedback to establish social distance during conversation. In order to gain rapport, people would most likely want to decrease social distance in order to achieve the connectedness and smoothness in conversation that is seen in human social interaction. Research in human-computer interaction has pursued these nonverbal behaviors through appropriate backchanneling, head nods, and gaze techniques, but still missing is attention to user emotional state, which can be detected through some of these nonverbal behaviors in voice.

Two methods for describing emotions are discrete and dimensional. Discrete emotions include anger, disgust, fear, joy, sadness, and surprise. Dimensional emotions use two or more components to describe affective state. More commonly used dimensions are Osgood *et al.*'s (1957) evaluation (a.k.a. valence), activity, and potency (a.k.a. power). Emotion research has had limited success at detecting discrete emotions, e.g. (D'Mello et al., 2008). In the tutoring domain, some have looked at appropriately responding to students based on their prosody in voice (Hollingsed and Ward, 2007). The difficulty of recognizing discrete emotions exists because humans typically show more subtle emotions in most real human-human interactions (Batliner et al., 2000). Forbes *et al.* (2004) had promising results by looking at a three-class set of emotions (positive, negative, neutral).

The intent of this research is to develop a method for detecting three dimensions of emotion from voice in order to build rapport. There is a possibility that using a dimensional approach will enable more accurate modeling of subtle emotions that exist in spontaneous human-human dialogs.

3 Hypotheses and Expected Contributions

The main hypothesis of this work is that a spoken dialog system with emotional intelligence will be more effective for gaining rapport than a spoken dialog system without emotional intelligence. In order to test this hypothesis, I will implement and evaluate a spoken dialog system. This system will

choose topics and content depending on user emotional state. The resulting system will advance the state of the art in three technologies: recognizing appropriate emotion, planning accordingly, and synthesizing appropriate emotion. The system will also demonstrate how to integrate these components.

In addition to choosing the correct content based on user emotional state, this research will investigate the effect of adding emotion to voice for rapport. The second hypothesis of the research is that expressing emotion in voice and choosing words, compared to expressing emotion only by choosing words, will be more effective for building rapport with users.

4 Approach

This section outlines the steps that have been completed and those that are still pending to accomplish the goals of the research.

4.1 Corpus Analysis and Baseline System

This work is based on a persuasive dialog corpus consisting of audio recordings of 10 interactions averaging 16 minutes in length. The corpus consists of rougly 1000 turns between a graduate coordinator and individual students. The graduate coordinator was a personable female staff member who was hired by the University to raise the graduate student count. The students were enrolled in an introductory Computer Science course and participated in the study as part of a research credit required for course completion. The students had little knowledge of the nature or value of graduate school and of the application process. Preliminary analysis of the corpus showed evidence of a graduate coordinator building rapport with students by using emotion.

A baseline system built using commercial state-of-the-art software was implemented based on the corpus (mainly the topics covered). Informal user comments about the baseline system helped determine missing features for automated rapport building technology. One salient feature that is missing is attention to emotion in voice. This confirmed the direction of this research.

This corpus was transcribed and annotated with dimensional emotions (activation, valence, and power) by two judges. Activation is defined as sounding ready to take action, valence is the amount of positive or negative sound in voice, and power is measured by the amount of dominance in voice. The dimensions are annotated numerically on scales from -100 to +100.

The following are examples taken from the corpus with annotated acoustic features.

- Example 1
  Grad.Coord(GC1): *So you're in the 1401 class?* [rising pitch]

  Subject(S1): *Yeah.* [higher pitch]

  GC2: *Yeah? How are you liking it so far?* [falling pitch]

  S2: *Um, it's alright, it's just the labs are kind of difficult sometimes, they can, they give like long stuff.* [slower speed]

  GC3: *Mm. Are the TAs helping you?*
  [lower pitch and slower speed]

  S3: *Yeah.* [rising pitch]

  GC4: *Yeah.* [rising pitch]

  S4: *They're doing a good job.*
  [normal pitch and normal speed]

  GC5: *Good, that's good, that's good.*
  [normal pitch and normal speed]

- Example 2
  GC6: *You're taking your first CS class huh.*
  [slightly faster voice]

  S5: *Yeah, I barely started.* [faster voice]

  GC7: *How are you liking it?*
  [faster voice, higher pitch]

  S6: *Uh, I like it a lot, actually, it's probably my favorite class.* [faster, louder]

  GC8: *Oh good.* [slower, softer]

S7: *That I'm taking right now yeah.* [slightly faster, softer]

GC9: *Oh that's good. That's exciting.* [slow and soft then fast and loud]

GC10: *Then you picked the right major you're not gonna change it three times like I did.* [faster, louder]

In the first example, the coordinator noticably raises her pitch at the end of her utterance. This is probably so that she can sound polite or interested. On line S2, the subject displays a falling pitch (which sounds negative) and the coordinator responds with a lower fundamental frequency and a slower speed. The subject sounds unsure by displaying a rising pitch in his answer (S3). The coordinator mirrors his response (GC4) and finally both interlocutors end with normal pitch and normal speed.

In the second example, the subject speaks faster than usual (S5). The coordinator compensates by adjusting her speed as well. From S6 through GC8, when the subject's voice gets louder, the coordinator's voice gets softer, almost as though she is backing off and letting the subject have some space. In GC9 the coordinator responds to the student's positive response (liking the class) and becomes immediately faster and louder.

A next step for the analysis is to determine the most expressive acoustic correlates for emotions. Informal auditory comparisons show some possible correlations (see Table 1). These correlations seem promising because many correspond with previous work (Schroder, 2004).

The emotion annotations of the two judges show that strategies for adaptive emotion responses can be extracted from the corpus. Communication Accomodation Theory states that interlocutors mirror nonverbal behaviors during interaction when attempting to decrease social distance. The coordinator's emotional responses were correlated with the student's emotional utterances to determine if emotional mirroring (matching student emotion and coordinator response) was present in the persuasive dialog corpus. This was the case in the valence dimension, which showed a correlation coefficient of 0.34.

Table 1: Informal analysis reveals acoustic correlates possibly associated with the dimensions of emotion

Dimension	High	Low
Activeness	Faster, more varied pitch, louder	Slower, less varied pitch, softer
Valence	Higher pitch throughout, laughter, speed up	Falling ending pitch, articulation of words, increasing loudness
Power	Faster, louder, falling ending pitch, articulation of word beginnings, longer vowels	Softer, higher pitch throughout, quick rise in pitch, smoother word connection

However, regarding power, there was an inverse relationship; if the student showed more power, the coordinator showed less (–0.30 correlation coefficient). Activation showed a small correlation coefficient (–0.14).

To realize a spoken dialog system that could model this responsive behavior, machine learning was used. The students' three emotion dimensions were taken as attributes and were used to predict the coordinators emotional responses using Bagging with REPTrees. Measuring the correlations between the predictions of the model and the actual values in the corpus revealed correlation coefficients of 0.347, 0.344, and 0.187 when predicting the coordinator's valence, power, and activation levels, respectively.

4.2 Full System

The full system will provide a means to evaluate whether emotion contributes to automated rapport building. This system will be based on several available technologies and previous research in spoken dialog systems.

Figure 2 shows the different components anticipated for the full system. The components that will be implemented for this research include emotion recognition, user modeling components, and text and emotion strategy databases. The other components will be based on available open source software packages. The implementation effort also in-

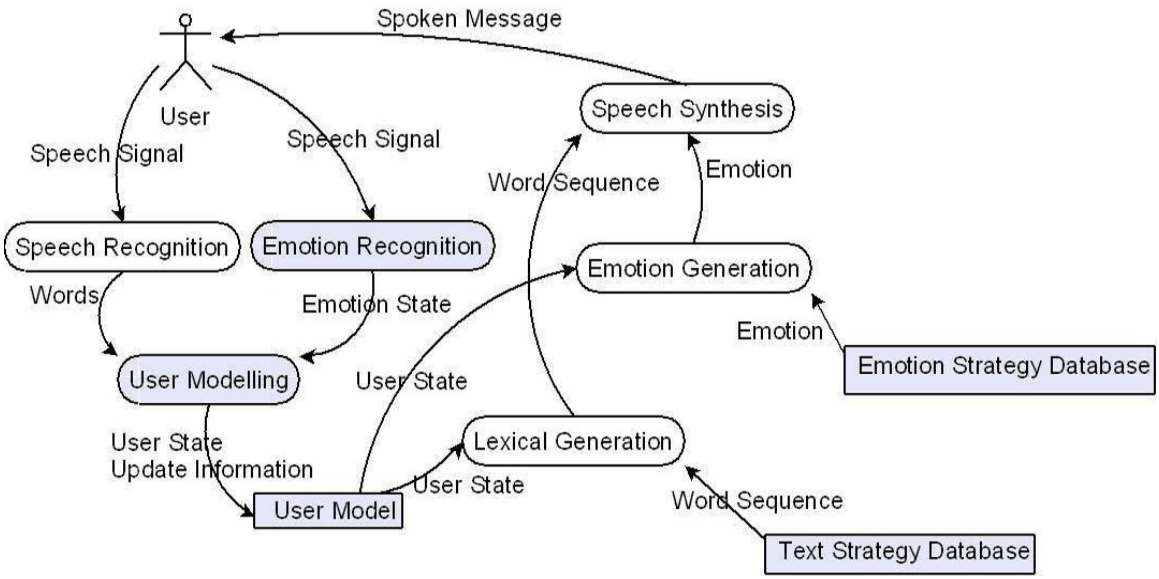

Figure 2: Full System Dataflow Diagram

cludes the integration of all components.

The following is a scenario that depicts how the full system will operate.

1. The system begins by saying "How are you doing today?"

2. The user says "I'm doing good" with a negative sounding voice.

3. The voice signal is then processed through the speech recognizer and emotion recognizer in parallel. The speech recognizer extracts words from the voice signal while the emotion recognizer extracts emotion.

4. This data is sent to the user modeling component which determines the immediate user state based only on the current emotion and the words spoken. In this scenario, the user's state will be negative even though the user stated otherwise.

5. This user state update information is then passed to the user model which updates the current user state. This component contains knowledge, beliefs and feelings of the user. Since there was no previous user state, the current emotion is set to negative. Stored in user knowledge will be the fact that the user was asked "How are you doing today?". Some information about the user's contradictory state is stored as user beliefs: stated good, but sounds negative.

6. Next, this information is used to select some predefined text from the lexical generation along with an associated emotion from the emotion strategy database (these two are done in parallel). Since the user's state is negative, the system may choose to ask another question such as "ok, do you have any concerns?" with a negative sounding voice (to mirror the valence dimension). In contrast, if the user was positive, the system may have chosen something similar to "great, let's get going then" with a highly positive voice.

7. Lastly, the text with corresponding emotion coloring is rendered to speech and played to the user by the speech synthesis component.

4.3 Evaluation

To achieve the final goal of determining whether emotion helps gain rapport, the final system described herein will be evaluated.

The final system will be configurable; it will allow for enabling emotion in voice (*voiced*) or disabling the emotions in voice (*not voiced*). In addition, there

will be a control configuration, perhaps one that will display a random emotion (*random*). A user study (hopefully within subjects) will be conducted that will ask users to interact with four versions of the system (baseline, *voiced*, *not voiced*, and *random*). A post-test questionnaire consisting of Likert scales will ask users how much rapport they felt with each version of the system. In addition, some objective metrics such as disfluency count and interaction time will be collected. This will help test the two hypotheses of this research. First, it is expected that subjects will have more rapport with the *not voiced* configuration than with the baseline system. The second hypothesis will be verified by determining if subjects have more rapport with the *voiced* than with the *not voiced* system. The *random* configuration will be used to determine whether the system's adaptive responses are better than random responses.

5 Broader Significance

This research addresses methods for gaining rapport as an important dimension of successful human-computer interaction, and one likely to be useful even for business-like dialogs. For example, building rapport with customers can decrease the number of disfluencies, which are currently a problem for speech recognizers. In addition, customer support systems will have the ability to tailor responses to decrease negative emotion.

Similarly, the learned rules for detecting emotion and responding appropriately could be used to train people how to more effectively gain rapport. Lastly, this work can supplement other rapport research that uses other forms of nonverbal behavior such as gaze and gestures seen especially in embodied conversational agents.

6 Acknowledgements

I would like to thank my advisor, Nigel Ward for his help. Also, I would like to thank Anais Rivera and Sue Walker for the collection of the persuasive dialog corpus and Jun Zheng for his help in fine tuning the baseline system. This work is supported in part by NSF grant IIS-0415150.

References

A. Batliner, K. Fischer, R. Huber, J. Spilker, and E. Nöth. 2000. Desperately Seeking Emotions or: Actors, Wizards, and Human Beings. In *ISCA Tutorial and Research Workshop (ITRW) on Speech and Emotion*. ISCA.

J. Cassell and T. Bickmore. 2003. Negotiated Collusion: Modeling Social Language and its Relationship Effects in Intelligent Agents. *User Modeling and User-Adapted Interaction*, 13(1):89–132.

T.L. Chartrand and J.A. Bargh. 1999. The chameleon effect: The perception-behavior link and social interaction. *Journal of Personality and Social Psychology*, 76(6):893–910.

S.K. D'Mello, S.D. Craig, A. Witherspoon, B. McDaniel, and A. Graesser. 2008. Automatic detection of learners affect from conversational cues. *User Modeling and User-Adapted Interaction*, 18(1):45–80.

K. Forbes-Riley and D. Litman. 2004. Predicting emotion in spoken dialogue from multiple knowledge sources. *Proc. Human Language Technology Conf. of the North American Chap. of the Assoc. for Computational Linguistics (HLT/NAACL)*.

J. Gratch, N. Wang, A. Okhmatovskaia, F. Lamothe, M. Morales, R.J. van der Werf, and L. Morency. 2007. Can Virtual Humans Be More Engaging Than Real Ones? *12th International Conference on Human-Computer Interaction*.

Tasha K. Hollingsed and Nigel G. Ward. 2007. A combined method for discovering short-term affect-based response rules for spoken tutorial dialog. *Workshop on Speech and Language Technology in Education (SLaTE)*.

J. O'Connor and J. Seymour. 1990. *Introducing neurolinguistic programming*. Mandala.

C.E. Osgood. 1957. *The Measurement of Meaning*. University of Illinois Press.

M. Schroder. 2004. Dimensional Emotion Representation as a Basis for Speech Synthesis with Non-extreme Emotions. *In Proceedings Workshop Affective Dialogue Systems*, 3068:209–220.

C.A. Shepard, H. Giles, and B.A. Le Poire. 2001. Communication accommodation theory. *The new handbook of language and social psychology*, pages 33–56.

Interactive Annotation Learning with Indirect Feature Voting

Shilpa Arora and Eric Nyberg
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{shilpaa,ehn}@cs.cmu.edu

Abstract

We demonstrate that a supervised annotation learning approach using structured features derived from tokens and prior annotations performs better than a bag of words approach. We present a general graph representation for automatically deriving these features from labeled data. Automatic feature selection based on class association scores requires a large amount of labeled data and direct voting can be difficult and error-prone for structured features, even for language specialists. We show that highlighted rationales from the user can be used for indirect feature voting and same performance can be achieved with less labeled data.We present our results on two annotation learning tasks for opinion mining from product and movie reviews.

1 Introduction

Interactive Annotation Learning is a supervised approach to learning annotations with the goal of minimizing the total annotation cost. In this work, we demonstrate that with additional supervision per example, such as distinguishing discriminant features, same performance can be achieved with less annotated data. Supervision for simple features has been explored in the literature (Raghavan et al., 2006; Druck et al., 2008; Haghighi and Klein, 2006). In this work, we propose an approach that seeks supervision from the user on structured features.

Features that capture the linguistic structure in text such as n-grams and syntactic patterns, referred to as *structured* features in this work, have been found to be useful for supervised learning of annotations. For example, Pradhan et al. (2004) show that using features like syntactic path from constituent to predicate improves performance of a semantic parser. However, often such features are "handcrafted" by domain experts and do not generalize to other tasks and domains. In this work, we propose a general graph representation for automatically extracting structured features from tokens and prior annotations such as part of speech, dependency triples, etc. Gamon (2004) shows that an approach using a large set of structured features and a feature selection procedure performs better than an approach that uses a few "handcrafted" features. Our hypothesis is that structured features are important for supervised annotation learning and can be automatically derived from tokens and prior annotations. We test our hypothesis and present our results for opinion mining from product reviews.

Deriving features from the annotation graph gives us a large number of very sparse features. Feature selection based on class association scores such as mutual information and chi-square have often been used to identify the most discriminant features (Manning et al., 2008). However, these scores are calculated from labeled data and they are not very meaningful when the dataset is small. Supervised feature selection, i.e. asking the user to vote for the most discriminant features, has been used as an alternative when the training dataset is small. Raghavan et al. (2006) and Druck et al. (2008) seek feedback on unigram features from the user for document classification tasks. Haghighi and Klein (2006) ask the user to suggest a few prototypes (examples) for each class and use those as features. These approaches ask the annotators to identify globally rel-

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 55–60,
Boulder, Colorado, June 2009.

evant features, but certain features are difficult to vote on without the context and may take on very different meanings in different contexts. Also, all these approaches have been demonstrated for unigram features and it is not clear how they can be extended straightforwardly to structured features.

We propose an indirect approach to interactive feature selection that makes use of highlighted rationales from the user. A *rationale* (Zaidan et al., 2007) is the span of text a user highlights in support of his/her annotation. Rationales also allow us to seek feedback on features in context. Our hypothesis is that with rationales, we can achieve same performance with lower annotation cost and we demonstrate this for opinion mining from movie reviews.

In Section 2, we describe the annotation graph representation and motivate the use of structured features with results on learning opinions from product reviews. In Section 3, we show how rationales can be used for identifying the most discriminant features for opinion classification with less training data. We then list the conclusions we can draw from this work, followed by suggestions for future work.

2 Learning with Structured Features

In this section, we demonstrate that structured features help in improving performance and propose a formal graph representation for deriving these features automatically.

2.1 Opinions and Structured Features

Unigram features such as tokens are not sufficient for recognizing all kinds of opinions. For example, a unigram feature *good* may seem useful for identifying opinions, however, consider the following two comments in a review: 1) *This camera has **good** features* and 2) *I did a **good** month's worth of research before buying this camera*. In the first example, the unigram *good* is a useful feature. However, in the second example, *good* is not complementing the camera and hence will mislead the classifier. Structured features such as part-of-speech, dependency relations etc. are needed to capture the language structure that unigram features fail to capture.

2.2 Annotation Graph and Features

We define the annotation graph as a quadruple: $G = (N, E, \Sigma, \lambda)$, where N is the set of nodes, E is the set of edges $E \subset N \times N$, $\Sigma = \Sigma_N \cup \Sigma_E$ is a set of labels for nodes and edges. λ is the labeling function $\lambda : N \cup E \rightarrow \Sigma$, that assigns labels to nodes and edges. In this work, we define the set of labels for nodes, Σ_N as tokens, part of speech and dependency annotations and set of labels for edges, Σ_E as relations, $\Sigma_E = \{\textit{leftOf}, \textit{parentOf}, \textit{restricts}\}$. The *leftOf* relation is defined between two adjacent nodes. The *parentOf* relation is defined between the dependency type and its attributes. For example, for the dependency triple 'nsubj_perfect_camera', there is a *parentOf* relation between the dependency type 'nsubj' and tokens 'perfect' and 'camera'. The *restricts* relation exists between two nodes a and b if their textual spans overlap completely and a restricts how b is interpreted. For a word with multiple senses the *restricts* relation between the word and its part of speech, restricts the way the word is interpreted, by capturing the sense of the word in the given context. The Stanford POS tagger (Toutanova and Manning, 2000) and the Stanford parser (Klein and Manning, 2003) were used to produce the part of speech and dependency annotations.

Features are defined as subgraphs, $G' = (N', E', \Sigma', \lambda')$ in the annotation graph G, such that $N' \subseteq N$, $E' \subset N' \times N'$ and $E' \subseteq E$, $\Sigma' = \Sigma'_N \cup \Sigma'_E$ where $\Sigma'_N \subseteq \Sigma_N$ and $\Sigma'_E \subseteq \Sigma_E$ and $\lambda' : N' \cup E' \rightarrow \Sigma'$. For a bag of words approach that only uses tokens as features, $\Sigma'_N = T$, where T is the token vocabulary and $E = \phi$ and $\Sigma_E = \phi$ (where ϕ is the null set). We define the $degree$ of a feature subgraph as the number of edges it contains. For example, the unigram features are the feature subgraphs with no edges i.e. $degree = 0$. $Degree - 1$ features are the feature subgraphs with two nodes and an edge. In this paper, we present results for feature subgraphs with $degree = 0$ and $degree = 1$.

Figure 1 shows the partial annotation graph for two comments discussed above. The feature subgraph that captures the opinion expressed in 1(a), can be described in simple words as "camera has features that are good". This kind of subject-object relationship with the same verb, between the 'camera' and what's being modified by 'good', is not present in the second example (1(b)). A slight modification of 1(b), *I did a month's worth of research before buying this **good** camera* does express an opinion about the camera. A bag of words approach that uses only unigram features will not be able to differ-

entiate between these two examples; structured features like dependency relation subgraphs can capture this linguistic distinction between the two examples.

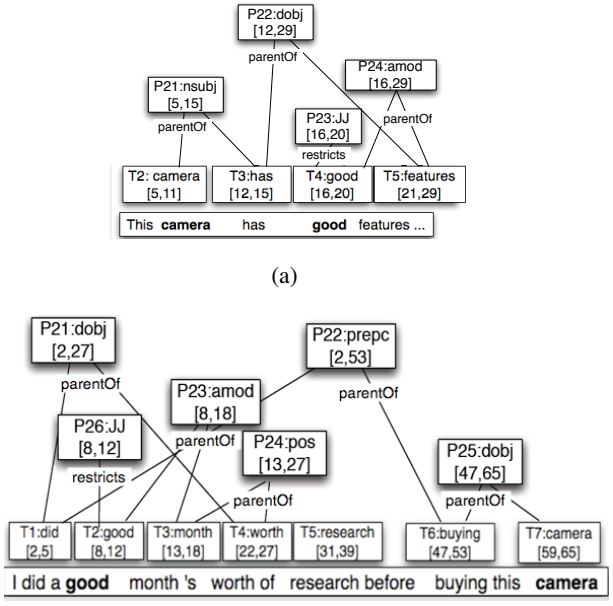

Figure 1: The figure shows partial annotation graphs for two examples. Only some of the nodes and edges are shown for clarity. Spans of nodes in brackets are the character spans.

2.3 Experiments and Results

The dataset we used is a collection of 244 Amazon's customer reviews (2962 comments) for five products (Hu and Liu, 2004). A review comment is annotated as an opinion if it expresses an opinion about an aspect of the product and the aspect is explicitly mentioned in the sentence. We performed 10-fold cross validation (CV) using the Support Vector Machine (SVM) classifier in MinorThird (Cohen, 2004) with the default linear kernel and chi-square feature selection to select the top 5000 features. As can be seen in Table 1, an approach using $degree - 0$ features, i.e. unigrams, part of speech and dependency triples together, outperforms using any of those features alone and this difference is significant. Using $degree - 1$ features with two nodes and an edge improves performance further. However, using $degree-0$ features in addition to $degree-1$ features does not improve performance. This suggests that when using higher degree features, we may leave out the features with lower degree that they subsume.

Features	Avg F1	Outperforms
unigram [uni]	65.74	pos,**dep**
pos-unigram [pos]	64	**dep**
dependency [dep]	63.18	-
degree-0 [deg-0]	67.77	**uni**,**pos**,**dep**
degree-1 [deg-1]	**70.56**	**uni**,**pos**,**dep**,**deg-0**, deg-*
(deg-0 + deg-1) [deg-*]	70.12	**uni**,**pos**,**dep**,**deg-0**

Table 1: The table reports the F-measure scores averaged over ten cross validation folds. The value in bold in the *Avg F1* column is the best performing feature combination. For each feature combination in the row, *outperforms* column lists the feature combinations it outperforms, with significant differences highlighted in bold (paired t-test with $p < 0.05$ considered significant).

3 Rationales & Indirect Feature voting

We propose an indirect feature voting approach that uses user-highlighted rationales to identify the most discriminant features. We present our results on Movie Review data annotated with rationales.

3.1 Data and Experimental Setup

The data set by Pang and Lee (2004) consists of 2000 movie reviews (1000-pos, 1000-neg) from the IMDb review archive. Zaidan et al. (2007) provide rationales for 1800 reviews (900-pos, 900-neg). The annotation guidelines for marking rationales are described in (Zaidan et al., 2007). An example of a *rationale* is: "the movie is **so badly put together** that even the most casual viewer may notice the **miserable pacing and stray plot threads**". For a test dataset of 200 reviews, randomly selected from 1800 reviews, we varied the training data size from 50 to 500 reviews, adding 50 reviews at a time. Training examples were randomly selected from the remaining 1600 reviews. During testing, information about rationales is not used.

We used tokens[1], part of speech and dependency triples as features. We used the KStem stemmer (Krovetz, 1993) to stem the token features. In order to compare the approaches at their best performing feature configuration, we varied the total number of features used, choosing from the set: {1000, 2000, 5000, 10000, 50000}. We used chi-square feature selection (Manning et al., 2008) and the SVM learner with default settings from the Minorthird package (Cohen, 2004) for these experiments. We compare the following approaches:

Base Training Dataset (BTD): We train a model from the labeled data with no feature voting.

[1]filtering the stop words using the stop word list: `http://www.cs.cmu.edu/~shilpaa/stop-words-ial-movie.txt`

Rationale *annotated* Training Dataset (RTD): We experimented with two different settings for indirect feature voting: 1) only using features that overlap with rationales ($RTD(1,0)$); 2) features from rationales weighted twice as much as features from other parts of the text ($RTD(2,1)$). In general, $R(i,j)$ describes an experimental condition where features from rationales are weighted i times and other features are weighted j times. In Minorthird, weighing a feature two times more than other features is equivalent to that feature occurring twice as much.

Oracle *voted* Training Data (OTD): In order to compare indirect feature voting to direct voting on features, we simulate the user's vote on the features with class association scores from a large dataset (all 1600 documents used for selecting training documents). This is based on the assumption that the class association scores, such as chi-square, from a large dataset can be used as a reliable discriminator of the most relevant features. This approach of simulating the oracle with large amount of labeled data has been used previously in feature voting (Raghavan et al., 2006).

3.2 Results and Discussion

In Table 2, we present the accuracy results for the four approaches described in the previous section. We compare the best performing feature configurations for three approaches - BTD, $RTD(1,0)$ and RTD (2,0). As can be seen, $RTD(1,0)$ always performs better than BTD. As expected, improvement with rationales is greater and it is significant when the training dataset is small. The performance of all approaches converge as the training data size increases and hence we only present results up to training dataset size of 500 examples in this paper.

Since our goal is to evaluate the use of rationales independently of how many features the model uses, we also compared the four approaches in terms of the accuracy averaged over five feature configurations. Due to space constraints, we do not include the table of results. On average $RTD(1,0)$ significantly outperforms BTD when the total training dataset is less than 350 examples. When the training data has fewer than 400 examples, $RTD(1,0)$ also significantly outperforms $RTD(2,1)$.

OTD with simulated user is an approximate upper bound for rationale based approaches. It tells us how far we are from direct supervision on structured features. On average, OTD significantly outperformed $RTD(1,0)$ for training data size of 100, 150, 400, 450 and 500 examples but not always. As can be seen from Table 2, difference between OTD and $RTD(1,0)$ reduces with more training data, since with more data and hence more rationales we get better feature coverage.

Results presented here show that for a given training dataset, we can boost the performance by ask-

#Ex	Approach	Number of Features				
		1000	**2000**	**5000**	**10000**	**50000**
50	OTD	**67.63**	66.30	62.90	52.17	55.03
	BTD	**58.10**	57.47	52.67	51.80	55.03
	RTD(1,0)*	55.43	55.93	**61.63**	61.63	61.63
	RTD(2,1)	**57.77**	57.53	52.73	52.30	56.33
100	OTD	**71.97**	71.07	70.27	69.37	64.33
	BTD	64.17	**64.43**	62.70	56.63	64.37
	RTD(1,0)*	65.43	63.27	65.13	**67.23**	67.23
	RTD(2,1)	**64.27**	63.93	62.47	56.10	63.77
150	OTD	73.83	**74.83**	74.20	74.00	63.83
	BTD	66.17	67.77	**68.60**	64.33	60.47
	RTD(1,0)*	69.30	68.30	67.27	71.30	**71.87**
	RTD(2,1)	**68.00**	67.07	68.43	63.57	58.90
200	OTD	74.83	**75.87**	75.70	75.10	56.97
	BTD	71.63	71.37	**72.57**	71.53	58.90
	RTD(1,0)	72.23	72.63	71.63	**73.80**	73.93
	RTD(2,1)	71.20	71.10	**73.03**	70.77	57.87
250	OTD	75.63	76.90	**77.70**	77.67	62.20
	BTD	72.60	73.57	74.73	**75.20**	58.93
	RTD(1,0)	73.00	73.57	73.57	74.70	**76.70**
	RTD(2,1)	72.87	73.90	74.63	**75.40**	57.43
300	OTD	76.57	77.67	**78.93**	78.43	68.17
	BTD	72.97	74.13	74.93	**76.57**	63.83
	RTD(1,0)	74.43	74.83	74.67	74.73	**77.67**
	RTD(2,1)	72.67	74.53	74.37	**76.53**	61.30
350	OTD	76.47	78.20	**80.20**	79.80	71.73
	BTD	74.43	74.30	74.73	**77.27**	66.80
	RTD(1,0)	75.07	76.20	75.80	75.20	**78.53**
	RTD(2,1)	74.63	75.70	74.80	**78.23**	64.93
400	OTD	77.97	78.93	80.53	**80.60**	75.27
	BTD	75.83	76.77	76.47	**78.93**	70.63
	RTD(1,0)	75.17	76.40	75.83	76.00	**79.23**
	RTD(2,1)	75.73	76.07	76.80	**78.50**	68.20
450	OTD	77.67	79.20	80.57	**80.73**	77.13
	BTD	75.73	76.80	77.80	**78.80**	74.37
	RTD(1,0)*	74.83	76.50	76.23	76.47	**80.40**
	RTD(2,1)	75.87	76.87	77.87	**78.87**	71.80
500	OTD	78.03	80.10	81.27	**81.67**	79.87
	BTD	75.27	77.33	79.37	**80.30**	75.73
	RTD(1,0)	75.77	77.63	77.47	77.27	**81.10**
	RTD(2,1)	75.83	77.47	79.50	**79.70**	74.50

Table 2: Accuracy performance for four approaches, five feature configurations and increasing training dataset size. Accuracy reported is averaged over five random selection of training documents for three randomly selected test datasets. The numbers in bold in a row represents the best performing feature configuration for a given approach and training dataset size. The approach in bold represents the best performing approach among BTD, $RTD(1,0)$ and $RTD(2,1)$ for a given training dataset size. '*' indicates significant improvement in performance over BTD (paired t-test with $p < 0.05$ considered significant).

ing the user to label rationales. However, there is an additional cost associated with the rationales. It is important to evaluate how much total annotation cost rationales can save us while achieving the desired performance. In Figure 2, we compare the number of training examples an approach needs to achieve a given level of performance. As can be seen, $RTD(1,0)$ needs fewer training examples to achieve the same performance as BTD. The difference is large initially when the total number of training examples is small (50 for $RTD(1,0)$ and 150 for BTD to achieve a performance between $66-67$).

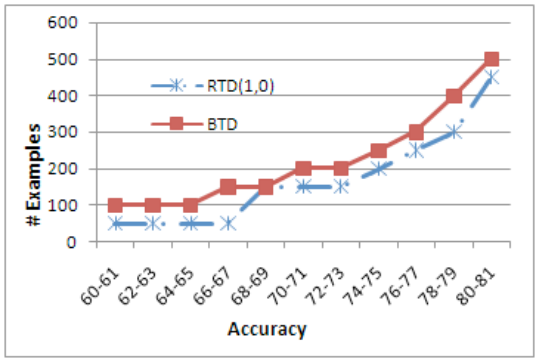

Figure 2: The Figure shows the number of examples needed by the two approaches, $RTD(1,0)$ and BTD, to achieve an accuracy in the given range.

Comparison with Zaidan et al. (2007): Zaidan et al. (2007) conclude that using only features from rationales performs worse than both: 1) using all the features in the documents, and 2) using features that do not overlap with the rationales. The results presented in this paper seem to contradict their results. However, they only experimented with unigram features and only one approach to using features from rationales, $RTD(1,0)$ and not $RTD(2,1)$. In order to compare our work directly with theirs, we experimented with an equivalent set of unigram features. In Table 3, we present the results using same number of total features (17744) as Zaidan et al. (2007). As can be seen from the table, when only unigram features are used, $RTD(2,1)$ outperforms BTD but $RTD(1,0)$ performs worse than BTD. Thus, our results are consistent with (Zaidan et al., 2007) i.e. using unigram features only from the rationales does not boost performance.

From Table 3, we also analyze the improvement in performance when part of speech and dependency features are used in addition to the unigram features i.e. using all $degree-0$ subgraph features. For $RTD(1,0)$, adding these features improves performance for all data sizes with significant improvement for dataset size of 300 and 500 examples. $RTD(1,0)$ also significantly outperforms BTD when all three features are used. For direct voting on features (OTD), a significant improvement with these structured features is seen when the training dataset size is greater than 200 examples. For BTD and $RTD(2,1)$ approaches, there is no significant improvement with these additional features. In the future, we plan to investigate further the benefit of using higher degree subgraph features for opinion mining from the movie review data.

#Ex	Approach	uni	uni-pos	uni-pos-dep
100	OTD	68.6	68.8	61.6
	BTD	68.6	68.8	52.2
	RTD(1,0)	68.2	68.1	**69.0***
	RTD(2,0)	**70.0**	67.0	51.7
200	OTD	73.6	73.8	75.3
	BTD	73.6	**73.8**	67.1
	RTD(1,0)	73.9	73.2	**73.9***
	RTD(2,0)	**75.3***	70.3	65.2
300	OTD	76.2	76.1	79.1
	BTD	76.2	**76.1**	73.7
	RTD(1,0)	75.0	74.9	**77.1***
	RTD(2,0)	**77.5***	73.3	74.8
400	OTD	77.4	76.8	79.9
	BTD	77.4	**76.8**	76.2
	RTD(1,0)	75.9	75.9	77.0
	RTD(2,0)	**78.0**	74.7	**77.7***
500	OTD	78.1	78.1	80.0
	BTD	78.1	**78.1**	78.4
	RTD(1,0)	76.3	76.2	77.6
	RTD(2,0)	**78.2**	75.4	**79.0**

Table 3: The Table reports accuracy for four approaches in a setting similar to (Zaidan et al., 2007). Accuracy reported is averaged over ten random selection of training documents for two randomly selected test datasets.The numbers in bold are the best among BTD, $RTD(1,0)$, $RTD(2,1)$ for a given feature combination. '*' highlights the significant improvement in performance over BTD (using paired t-test, with $p < 0.05$ considered significant).

Comparing ranking of features: We also compared the features that the rationales capture to what the oracle will vote for as the most relevant features. Features are ranked based on chi-square scores used in feature selection. We compare the ranked list of features from $RTD(1,0)$, BTD and OTD and use a weighted F-measure score for evaluating the top 100 ranked features by each approach. This measure is inspired by the *Pyramid* measure used in Summarization (Nenkova and Passonneau, 2004). Instead of using counts in calculating F-measure, we used the chi-square score assigned to the features by the oracle dataset, in order to give more weight to the more discriminant features. As can be seen from

Table 4, $RTD(1,0)$ outperforms BTD in capturing the important features when the datasize set is small (< 300) and this difference is significant. Beyond 300 examples, as the data size increases, BTD outperforms $RTD(1,0)$. This implies that the rationales alone are able to capture the most relevant features when the dataset is small.

	100	200	300	400	500	600	700
RO	**47.70**	**53.80**	**57.68**	59.54	62.13	60.86	61.56
TO	31.22	44.43	52.98	60.57	64.61	**67.10**	**70.39**

Table 4: Weighted F-measure performance comparison of ranked list of features from $RTD(1,0)$ & $OTD(RO)$ and BTD & $OTD(TO)$. Results are averaged over ten random selections of the training data for a randomly selected test dataset. Significant differences are highlighted in bold (paired t-test with $p < 0.05$ considered significant).

4 Conclusion and Future Work

In this work, we demonstrated that using structured features boosts performance of supervised annotation learning. We proposed a formal annotation graph representation that can be used to derive these features automatically. However, the space of possible feature subgraphs can grow very large with more prior annotations. Standard feature selection techniques based on class association scores are less effective when the dataset is small. Feature voting from the user for identifying the relevant features is limited to simple features. Supplementary input from the user in terms of highlighted rationales can be used instead to prune the feature space. The proposed approach is general and can be applied to a variety of problems and features.

In this work, we presented our results with $degree - 0$ and $degree - 1$ feature subgraphs. We will extend our algorithm to automatically extract higher degree features from the annotation graph. For the rationale annotated training data ($RTD(i,j)$), we experimented with two possible values for i and j. We aim to learn these weights empirically using a held out dataset. Rationales are associated with an additional cost per example and hence two approaches, with and without the rationales, are not directly comparable in terms of the number of examples. In the future, we will conduct an annotation experiment with real users to evaluate the usefulness of rationales in terms of clock time.

Acknowledgments

We would like to thank Dr. Carolyn P. Rose for her help with statistical analysis of the results. We would also like to thank all the anonymous reviewers for their helpful comments.

References

Cohen W. *Minorthird: Methods for Identifying Names and Ontological Relations in Text using Heuristics for Inducing Regularities from Data.* 2004. (`http://minorthird.sourceforge.net/`).

Druck G., Mann G. and McCallum A. *Learning from labeled features using generalized expectation criteria.* In Proceedings of the ACM SIGIR, 2008.

Michael Gamon. 2004. *Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis.* In Proceedings of COLING, 2005.

Haghighi A. and Klein D. *Prototype-driven learning for sequence models.* In Proceedings of the NAACL HLT 2006.

Minqing Hu and Bing Liu. 2004. *Mining and Summarizing Customer Reviews*. In Proc. of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.

Klein D. and Manning C. *Accurate Unlexicalized Parsing.* In Proceedings of ACL 2003.

Krovetz R. *Viewing Morphology as an Inference Process.* `http://ciir.cs.umass.edu/pubfiles/ir-35.pdf`

Manning C., Raghavan P. and Schütze H. *Introduction to Information Retrieval.* Cambridge University Press. 2008.

Nenkova A. and Passonneau R. *Evaluating Content Selection In Summarization: The Pyramid Method.* In Proceedings of HLT-NAACL 2004.

Pang B. and Lee L. *"A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts"* In Proceedings of the ACL, 2004.

Sameer S. Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin, Daniel Jurafsky. 2004. *Shallow Semantic Parsing using Support Vector Machines.* In Proceedings of HLT/NAACL-2004,Boston, MA, May 2-7, 2004

Raghavan H., Madani O. and Jones R. *Active Learning with Feedback on Both Features and Instances.* Journal of Machine Learning Research, 2006.

Toutanova K. and Manning C. *Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger.* In Proceedings of EMNLP/VLC-2000.

Zaidan O., Eisner J. and Piatko C. *Using "annotator rationales" to improve machine learning for text categorization.* In Proceedings of NAACL-HLT 2007.

Zaidan O. and Eisner J. *Modeling Annotators: A Generative Approach to Learning from Annotator Rationales.* In Proceedings of EMNLP 2008.

Loss-Sensitive Discriminative Training of Machine Transliteration Models

Kedar Bellare
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003, USA
kedarb@cs.umass.edu

Koby Crammer
Department of Computer Science
University of Pennsylvania
Philadelphia, PA 19104, USA
crammer@cis.upenn.edu

Dayne Freitag
SRI International
San Diego, CA 92130, USA
dayne.freitag@sri.com

Abstract

In machine transliteration we transcribe a name across languages while maintaining its phonetic information. In this paper, we present a novel sequence transduction algorithm for the problem of machine transliteration. Our model is discriminatively trained by the MIRA algorithm, which improves the traditional Perceptron training in three ways: (1) It allows us to consider k-best transliterations instead of the best one. (2) It is trained based on the ranking of these transliterations according to user-specified loss function (Levenshtein edit distance). (3) It enables the user to tune a built-in parameter to cope with noisy non-separable data during training. On an Arabic-English name transliteration task, our model achieves a relative error reduction of 2.2% over a perceptron-based model with similar features, and an error reduction of 7.2% over a statistical machine translation model with more complex features.

1 Introduction and Related Work

Proper names and other technical terms are frequently encountered in natural language text. Both machine translation (Knight and Graehl, 1997) and cross-language information retrieval (Jeong et al., 1999; Virga and Khudanpur, 2003; Abdul-Jaleel and Larkey, 2003) can benefit by explicitly translating such words from one language into another. This approach is decidedly better than treating them uniformly as out-of-vocabulary tokens. The goal of machine *transliteration* is to translate words between alphabets of different languages such that they are phonetically equivalent.

Given a source language sequence $\mathbf{f} = f_1 f_2 \dots f_m$ from an alphabet $\mathcal{F}$, we want to produce a target language sequence $\mathbf{e} = e_1 e_2 \dots e_n$ in the alphabet $\mathcal{E}$ such that it maximizes some score function $s(\mathbf{e}, \mathbf{f})$,

$$\mathbf{e} = \arg\max_{\mathbf{e}'} s(\mathbf{e}', \mathbf{f}).$$

Virga and Khudanpur (2003) model this scoring function using a separate *translation* and *language* model, that is, $s(\mathbf{e}, \mathbf{f}) = Pr(\mathbf{f}|\mathbf{e})Pr(\mathbf{e})$. In constrast, Al-Onaizan and Knight (2002) directly model the translation probability $Pr(\mathbf{e}|\mathbf{f})$ using a log-linear combination of several individually trained phrase and character-based models. Others have treated transliteration as a phrase-based transduction (Sherif and Kondrak, 2007). All these approaches are adaptations of statistical models for machine translation (Brown et al., 1994). In general, the parameters of the scoring function in such approaches are trained generatively and do not utilize complex features of the input sequence pairs.

Recently, there has been interest in applying discriminatively-trained sequence alignment models to many real-world problems. McCallum et al. (2005) train a conditional random field model to discriminate between matching and non-matching string pairs treating alignments as latent. Learning accurate alignments in this model requires finding "close" non-match pairs which can be a challenge. A similar conditional latent-variable model has been applied to the task of lemmatization and generation of morphological forms (Dreyer et al., 2008).

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 61–65,
Boulder, Colorado, June 2009.

Zelenko and Aone (2006) model transliteration as a structured prediction problem where the letter e_i is predicted using local and global features derived from $e_1 e_2 \ldots e_{i-1}$ and $\mathbf{f}$. Bergsma and Kondrak (2007) address cognate identification by training a SVM classification model using phrase-based features obtained from a Levenshtein alignment. Both these models do not learn alignments that is needed to obtain high performance on transliteration tasks. Freitag and Khadivi (2007) describe a discriminatively trained sequence alignment model based on averaged perceptron, which is closely related to the method proposed in this paper.

Our approach improves over previous directions in two ways. First, our system produces better k-best transliterations than related approaches by training on multiple hypotheses ranked according to a user-specified loss function (Levenshtein edit distance). Hence, our method achieves a 19.2% error reduction in 5-best performance over a baseline only trained with 1-best transliterations. This is especially helpful when machine transliteration is part of a larger machine translation or information retrieval pipeline since additional sentence context can be used to choose the best among top-K transliterations. Second, our training procedure accounts for noise and non-separability in the data. Therefore, our transliteration system would work well in cases where person names were misspelled or in cases in which a single name had many reasonable translations in the foreign language.

The training algorithm we propose in this paper is based on the K-best MIRA algorithm which has been used earlier in structured prediction problems (McDonald et al., 2005a; McDonald et al., 2005b). Our results demonstrate a significant improvement in accuracy of 7.2% over a statistical machine translation (SMT) system (Zens et al., 2005) and of 2.2% over a perceptron-based edit model (Freitag and Khadivi, 2007).

2 Sequence Alignment Model

Let $\mathbf{e} = e_1 e_2 \ldots e_n$ and $\mathbf{f} = f_1 f_2 \ldots f_m$ be sequences from the target alphabet $\mathcal{E}$ and source alphabet $\mathcal{F}$ respectively. Let $\mathbf{a} = a_1 a_2 \ldots a_l$ be a sequence of alignment operations needed to convert $\mathbf{f}$ into $\mathbf{e}$. Each alignment operation either appends a letter to the end of the source sequence, the target sequence or both sequences. Hence, it is a member of the cross-product $a_k \in \mathcal{E} \cup \{\epsilon\} \times \mathcal{F} \cup \{\epsilon\} \backslash \{(\epsilon, \epsilon)\}$, where ϵ is the null character symbol. Let $\mathbf{a}_1^k = a_1 a_2 \ldots a_k$ denote the sequence of first k alignment operations. Similarly $\mathbf{e}_1^k$ and $\mathbf{f}_1^k$ are prefixes of $\mathbf{e}$ and $\mathbf{f}$ of length k.

We define the scoring function between a word and its transliteration to be the a maximum over all possible alignment sequences $\mathbf{a}$,

$$s(\mathbf{e}, \mathbf{f}) = \max_{\mathbf{a}} s(\mathbf{a}, \mathbf{e}, \mathbf{f}) \ ,$$

where the score of a specific alignment $\mathbf{a}$ between two words is given by a linear relation,

$$s(\mathbf{a}, \mathbf{e}, \mathbf{f}) = \mathbf{w} \cdot \Phi(\mathbf{a}, \mathbf{e}, \mathbf{f}),$$

for a parameter vector $\mathbf{w}$ and a feature vector $\Phi(\mathbf{a}, \mathbf{e}, \mathbf{f})$. Furthermore, let $\Phi(\mathbf{a}, \mathbf{e}, \mathbf{f}) = \sum_{k=1}^{l} \phi(a_k, \mathbf{e}, i, \mathbf{f}, j)$ be the sum of feature vectors associated with individual alignment operations. Here i, j are positions in sequences $\mathbf{e}, \mathbf{f}$ after performing operations $\mathbf{a}_1^k$. For fixed sequences $\mathbf{e}$ and $\mathbf{f}$ the function $s(\mathbf{e}, \mathbf{f})$ can be efficiently computed using a dynamic programming algorithm,

$$s(\mathbf{e}_1^i, \mathbf{f}_1^j) = \max \begin{cases} s(\mathbf{e}_1^{i-1}, \mathbf{f}_1^j) + \mathbf{w} \cdot \phi(\langle e_i, \epsilon \rangle, \mathbf{e}, i, \mathbf{f}, j) \\ s(\mathbf{e}_1^i, \mathbf{f}_1^{j-1}) + \mathbf{w} \cdot \phi(\langle \epsilon, f_j \rangle, \mathbf{e}, i, \mathbf{f}, j) \\ s(\mathbf{e}_1^{i-1}, \mathbf{f}_1^{j-1}) + \mathbf{w} \cdot \phi(\langle e_i, f_j \rangle, \mathbf{e}, i, \mathbf{f}, j). \end{cases} \quad (1)$$

Given a source sequence $\mathbf{f}$ computing the best scoring target sequence $\mathbf{e} = \arg\max_{\mathbf{e}'} s(\mathbf{e}', \mathbf{f})$ among all possible sequences $\mathcal{E}^*$ requires a beam search procedure (Freitag and Khadivi, 2007). This procedure can also be used to produce K-best target sequences $\{\mathbf{e}'_1, \mathbf{e}'_2, \ldots, \mathbf{e}'_K\}$ such that $s(\mathbf{e}'_1, \mathbf{f}) \geq s(\mathbf{e}'_2, \mathbf{f}) \geq \ldots \geq s(\mathbf{e}'_K, \mathbf{f})$.

In this paper, we employ the same features as those used by Freitag and Khadivi (2007). All local feature functions $\phi(a_k, \mathbf{e}, i, \mathbf{f}, j)$ are conjunctions of the alignment operation a_k and forward or backward-looking character m-grams in sequences $\mathbf{e}$ and $\mathbf{f}$ at positions i and j respectively. For the source sequence $\mathbf{f}$ both forward and backward-looking m-gram features are included. We restrict the m-gram features in our target sequence $\mathbf{e}$ to only

be backward-looking since we do not have access to forward-looking m-grams during beam-search. An order M model is one that uses m-gram features where $m = 0, 1, \ldots M$.

Our training algorithm takes as input a data set $\mathcal{D}$ of source-target transliteration pairs and outputs a parameter vector $\mathbf{u}$. The algorithm pseudo-code appears in Fig. (1). In the algorithm, the function $\mathcal{L}(\mathbf{e}', \mathbf{e})$ defines a loss incurred by predicting $\mathbf{e}'$ instead of $\mathbf{e}$. In most structured prediction problems, the targets are of equal length and in such cases the Hamming loss function can be used. However, in our case the targets may differ in terms of length and thus we use the Levenshtein edit distance (Levenshtein, 1966) with unit costs for insertions, deletions and substitutions. Since the targets are both in the same alphabet $\mathcal{E}$ this loss function is well-defined. The user also supplies three paramters: (1) T - the number of training iterations (2) K - the number of best target hypotheses used (3) C - a complexity parameter. A low C is useful if the data is non-separable and noisy.

The final parameter vector $\mathbf{u}$ returned by the algorithm is the average of the intermediate parameter vectors produced during training. We find that averaging helps to improve performance. At test time, we use the beam search procedure to produce K-best hypotheses using the parameter vector $\mathbf{u}$.

Input parameters

Training Data	$\mathcal{D}$
Complexity parameter	$C > 0$
Number of epochs	T

Initialize $\mathbf{w}_0 = \mathbf{0}$ (zero vector) ; $\tau = 0$; $\mathbf{u} = \mathbf{0}$

Repeat T times:

For Each $(\mathbf{e}, \mathbf{f}) \in \mathcal{D}$:

1. $\mathbf{a} = \arg\max_{\hat{\mathbf{a}}} \mathbf{w}_\tau \cdot \Phi(\hat{\mathbf{a}}, \mathbf{e}, \mathbf{f})$ (Find best scoring alignment between $\mathbf{e}$ and $\mathbf{f}$ using dynamic programming)
2. Generate a list of K-best target hypotheses $\{\mathbf{e}'_1, \mathbf{e}'_2, \ldots, \mathbf{e}'_K\}$ given the current parameters $\mathbf{w}_\tau$. Let the corresponding alignments for the targets be $\{\mathbf{a}'_1, \mathbf{a}'_2, \ldots, \mathbf{a}'_K\}$.
3. Set $\mathbf{w}_{\tau+1}$ to be the solution of :

$$\min_{\mathbf{w}} \tfrac{1}{2}||\mathbf{w} - \mathbf{w}_\tau||^2 + C \sum_{k=1}^{K} \xi_k$$
subject to (for $k = 1 \ldots K$) :
$$\mathbf{w} \cdot (\Phi(\mathbf{a}, \mathbf{e}, \mathbf{f}) - \Phi(\mathbf{a}'_k, \mathbf{e}'_k, \mathbf{f})) \geq \mathcal{L}(\mathbf{e}, \mathbf{e}'_k) - \xi_k$$
$$\xi_k \geq 0$$

4. $\mathbf{u} \leftarrow \mathbf{u} + \mathbf{w}_{\tau+1}$
5. $\tau \leftarrow \tau + 1$

Output Scoring function $s(\mathbf{a}, \mathbf{e}, \mathbf{f}) = \mathbf{u} \cdot \Phi(\mathbf{a}, \mathbf{e}, \mathbf{f})$

Figure 1: The k-best MIRA algorithm for discriminative learning of transliterations.

3 Experimental Results

We apply our model to the real-world Arabic-English name transliteration task on a data set of 10,084 Arabic names from the LDC. The data set consists of Arabic names in an ASCII-based alphabet and its English rendering. Table 1 shows a few examples of Arabic-English pairs in our data set. We use the same training/development/testing (8084/1000/1000) set as the one used in a previous benchmark study (Freitag and Khadivi, 2007). The development and testing data were obtained by randomly removing entries from the training data. The absence of short vowels (e.g. "a" in ⟨NB"I, nab'i⟩), doubled consonants (e.g. "ww" in ⟨FWAL, fawwal⟩) and other diacritics in Arabic make the transliteration a hard problem. Therefore, it is hard to achieve perfect accuracy on this data set.

Arabic	English
NB"I	nab'i
HNBLI	hanbali
FRIFI	furayfi
MLKIAN	malikian
BI;ANT	bizant
FWAL	fawwal
OALDAWI	khalidawi
BUWUI	battuti
H;?	hazzah

Table 1: Examples of Arabic names in the ASCII alphabet and their English transliterations.

For training, we set $K = 20$ best hypotheses and $C = 1.0$ and run the algorithm for $T = 10$ epochs. To evaluate our algorithm, we generate 1-best (or 5-best) hypotheses using the beam search procedure and measure accuracy as the percentage of instances in which the target sequence $\mathbf{e}$ is one of the 1-best (or 5-best) targets. The input features are based on character m-grams for $m = 1, 2, 3$. Unlike previ-

ous generative transliteration models, no additional language model feature is used.

We compare our model against a state-of-the-art statistical machine translation (SMT) system (Zens et al., 2005) and an averaged perceptron edit model (PTEM) with identical features (Freitag and Khadivi, 2007). The SMT system directly models the posterior probability $Pr(\mathbf{e}|\mathbf{f})$ using a log-linear combination of several sub-models: a character-based phrase translation model, a character-based lexicon model, a character penalty and a phrase penalty. In the PTEM model, the update rule only considers the best target sequence and modifies the parameters $\mathbf{w}_{\tau+1} = \mathbf{w}_\tau + \Phi(\mathbf{a}, \mathbf{e}, \mathbf{f}) - \Phi(\mathbf{a}', \mathbf{e}', \mathbf{f})$ if the score $s(\mathbf{e}', f) \geq s(\mathbf{e}, f)$.

Model (*train+dev*)	*1-best*	*5-best*
SMT	0.528	0.824
PTEM	0.552	0.803
MIRA	0.562	0.841

Table 2: The 1-best and 5-best accuracy of different models on the Arabic-English transliteration task. At 95% confidence level, MIRA/PTEM outperform the SMT model in 1-best accuracy and MIRA outperforms PTEM/SMT in 5-best accuracy.

Table 2 shows the *1-best* and *5-best* accuracy of each model trained on the combined *train+dev* data set. All the models are evaluated on the same *test* set. Both MIRA and PTEM algorithms outperform the SMT model in terms of *1-best accuracy*. The differences in accuracy are significant at 95% confidence level, using the bootstrapping method for hypothesis testing. The difference in *1-best* performance of MIRA and PTEM is not significant. At *5-best*, the MIRA model outperforms both SMT and PTEM model. We conjecture that using the problem-specific Levenshtein loss function helps filter bad target sequences from the K-best outputs during training.

In a second experiment we studied the effect of changing C on the performance of the algorithm. We ran the algorithm with the above settings, except varying the value of the complexity parameter to one of 7 values in the range $C = 0.00001, 0.0001, \ldots, 0.1, 1.0$, training only using the *train* set, and evaluating the resulting model on the *test* set. The results are summarized in Table 3. The entry marked with a star $*$ indicates the model that achieved the best performance on the *dev* set for a particular choice of evaluation measure (1-best or 5-best). We find that changing C does have an effect on model performance. As the value of C decreases, the performance at lower ranks improves: $C = 0.01$ is good for 5-best accuracy and $C = 0.001$ for 20-best accuracy (not in table). As C is further reduced, a greater number of iterations are needed to converge. In our model, where the alignments are not observed but inferred during training, we find that making small incremental updates makes our algorithm more robust. Indeed, setting $C = 0.01$ and training on the *train+dev* set improves 5-best performance of our model from 0.841 to 0.861. Hence, the choice of C is important.

Model (*train*)	*1-best*	*5-best*
$C = 1.0$	0.545^*	0.832
$C = 0.5$	0.548^*	0.83
$C = 0.2$	0.549^*	0.834
$C = 0.01$	0.545	0.852^*
$C = 0.001$	0.518	0.843
$C = 0.0001$	0.482	0.798
$C = 0.00001$	0.476	0.798

Table 3: The effect of varying model parameter C on *1,5-best* accuracy on the *test* set. All the models are trained with Levenshtein loss and 20-best targets. The superscript $*$ indicates the models that achieved the greatest performance on the *dev* set for a particular column.

4 Conclusions and Future Work

We have shown a significant improvement in accuracy over state-of-the-art transliteration models by taking into consideration the ranking of multiple hypotheses (top-K) by Levenshtein distance, and making the training algorithm robust to noisy non-separable data. Our model does consistently well at high ($K = 1$) *and* low ranks ($K = 5$), and can therefore be used in isolation or in a pipelined system (e.g. machine translation or cross-language information retrieval) to achieve better performance. In a pipeline system, more features of names around proper nouns and previous mentions of the name can be used to improve scoring of K-best outputs.

In our experiments, the Levenshtein loss function uses only unit costs for edit operations and is not specifically tuned towards our application. In future work, we may imagine penalizing insertions and deletions higher than substitutions and other non-uniform schemes for better transliteration performance. Our K-best framework can also be easily extended to cases where one name has multiple foreign translations that are equally likely.

References

Nasreen Abdul-Jaleel and Leah S. Larkey. 2003. Statistical transliteration for English-Arabic cross language information retrieval. In *CIKM '03*, pages 139–146, New York, NY, USA. ACM.

Yaser Al-Onaizan and Kevin Knight. 2002. Machine transliteration of names in arabic text. In *Proceedings of the ACL-02 Workshop on Computational Approaches to Semitic Languages*, pages 1–13.

Shane Bergsma and Greg Kondrak. 2007. Alignment-based discriminative string similarity. In *ACL*, pages 656–663, June.

Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1994. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Markus Dreyer, Jason Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1080–1089, Honolulu, Hawaii, October. Association for Computational Linguistics.

Dayne Freitag and Shahram Khadivi. 2007. A sequence alignment model based on the averaged perceptron. In *EMNLP-CoNLL*, pages 238–247.

K.S. Jeong, S. H. Myaeng, J.S. Lee, and K.-S. Choi. 1999. Automatic identification and back-transliteration of foreign words for information retrieval. *Information Processing and Management*, 35:523–540.

Kevin Knight and Jonathan Graehl. 1997. Machine transliteration. In Philip R. Cohen and Wolfgang Wahlster, editors, *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 128–135, Somerset, New Jersey. Association for Computational Linguistics.

Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Andrew McCallum, Kedar Bellare, and Fernando Pereira. 2005. A conditional random field for discriminatively-trained finite-state string edit distance. In *UAI*.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Flexible text segmentation with structured multilabel classification. In *HLT-EMNLP*, pages 987–994, Vancouver, BC, Canada, October. Association for Computational Linguistics.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005b. Online large-margin training of dependency parsers. In *ACL*, pages 91–98, Ann Arbor, Michigan, June.

Tarek Sherif and Grzegorz Kondrak. 2007. Substring-based transliteration. In *ACL*, pages 944–951, Prague, Czech Republic, June. Association for Computational Linguistics.

Paola Virga and Sanjeev Khudanpur. 2003. Transliteration of proper names in cross-lingual information retrieval. In *Proceedings of the ACL 2003 workshop on Multilingual and Mixed-language Named Entity Recognition*, pages 57–64, Morristown, NJ, USA. Association for Computational Linguistics.

Dmitry Zelenko and Chinatsu Aone. 2006. Discriminative methods for transliteration. In *EMNLP*, pages 612–617, Sydney, Australia, July. Association for Computational Linguistics.

R. Zens, O. Bender, S. Hasan, S. Khadivi, E. Matusov, J. Xu, Y. Zhang, and H. Ney. 2005. The RWTH Phrase-based Statistical Machine Translation System. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, Pittsburgh, PA, USA.

Syntactic Tree-based Relation Extraction Using a Generalization of Collins and Duffy Convolution Tree Kernel

Mahdy Khayyamian
Sharif University of Technology
khayyamian@ce.sharif.edu

Seyed Abolghasem Mirroshandel
Sharif University of Technology
mirroshandel@ce.sharif.edu

Hassan Abolhassani
Sharif University of Technology
abolhassani@sharif.edu

Abstract

Relation extraction is a challenging task in natural language processing. Syntactic features are recently shown to be quite effective for relation extraction. In this paper, we generalize the state of the art syntactic convolution tree kernel introduced by Collins and Duffy. The proposed generalized kernel is more flexible and customizable, and can be conveniently utilized for systematic generation of more effective application specific syntactic sub-kernels. Using the generalized kernel, we will also propose a number of novel syntactic sub-kernels for relation extraction. These kernels show a remarkable performance improvement over the original Collins and Duffy kernel in the extraction of ACE-2005 relation types.

1 Introduction

One of the contemporary demanding NLP tasks is information extraction, which is the procedure of extracting structured information such as entities, relations, and events from free text documents. As an information extraction sub-task, semantic relation extraction is the procedure of finding predefined semantic relations between textual entity mentions. For instance, assuming a semantic relation with type *Physical* and subtype *Located* between an entity of type *Person* and another entity of type *Location*, the sentence "*Police arrested Mark at the airport last week.*" conveys two mentions of this relation between "Mark" and "airport" and also between "police" and "airport" that can be shown in the following format.

Phys.Located(Mark, airport)
Phys.Located(police, airport)

Relation extraction is a key step towards question answering systems by which vital structured data is acquired from underlying free text resources. Detection of protein interactions in biomedical corpora (Li et al., 2008) is another valuable application of relation extraction.

Relation extraction can be approached by a standard classification learning method. We particularly use SVM (Boser et al., 1992; Cortes and Vapnik, 1995) and kernel functions as our classification method. A kernel is a function that calculates the inner product of two transformed vectors of a high dimensional feature space using the original feature vectors as shown in eq. 1.

$$K(X_i, X_j) = \phi(X_i).\phi(X_j) \quad (1)$$

Kernel functions can implicitly capture a large amount of features efficiently; thus, they have been widely used in various NLP tasks.

Various types of features have been exploited so far for relation extraction. In (Bunescu and Mooney, 2005b) *sequence of words* features are utilized using a sub-sequence kernel. In (Bunescu and Mooney, 2005a) *dependency graph* features are exploited, and in (Zhang et al., 2006a) *syntactic features* are employed for relation extraction. Although in order to achieve the best performance, it is necessary to use a proper combination of these features (Zhou et al., 2005), in this paper, we will concentrate on how to better capture the *syntactic features* for relation extraction.

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 66–71,
Boulder, Colorado, June 2009.

In CD'01 (Collins and Duffy, 2001) a convolution syntactic tree kernel is proposed that generally measures the syntactic similarity between parse trees. In this paper, a generalized version of CD'01 convolution tree kernel is proposed by associating generic weights to the nodes and sub-trees of the parse tree. These weights can be used to incorporate domain knowledge into the kernel and make it more flexible and customizable. The generalized kernel can be conveniently used to generate a variety of syntactic sub-kernels (including the original CD'01 kernel), by adopting appropriate weighting mechanisms.

As a result, in this paper, novel syntactic sub-kernels are generated from the generalized kernel for the task of relation extraction. Evaluations demonstrate that these kernels outperform the original CD'01 kernel in the extraction of ACE-2005 main relation types

The remainder of this paper is structured as follows. In section 2, the most related works are briefly reviewed. In section 3, CD'01 tree kernel is described. The proposed generalized convolution tree kernel is explained in section 4 and its produced sub-kernels for relation extraction are illustrated in section 5. The experimental results are discussed in section 6. Our work is concluded in section 7 and some possible future works are presented in section 8.

2 Related Work

In (Collins and Duffy, 2001), a convolution parse tree kernel has been introduced. This kernel is generally designed to measure syntactic similarity between parse trees and is especially exploited for parsing English sentences in their paper. Since then, the kernel has been widely used in different applications such as semantic role labeling (Moschitti, 2006b) and relation extraction (Zhang et al., 2006a; Zhang et al., 2006b; Zhou et al., 2007; Li et al. 2008).

For the first time, in (Zhang et al., 2006a), this convolution tree kernel was used for relation extraction. Since the whole syntactic parse tree of the sentence that holds the relation arguments contains a plenty of misleading features, several parse tree portions are studied to find the most feature-rich portion of the syntactic tree for relation extraction, and Path-Enclosed Tree (PT) is finally found to be the best performing tree portion. PT is a portion of parse tree that is enclosed by the shortest path between the two relation arguments. Moreover, this tree kernel is combined with an entity kernel to form a reportedly high quality composite kernel in (Zhang et al., 2006b).

3 CD'01 Convolution Tree Kernel

In (Collins and Duffy, 2001), a convolution tree kernel has been introduced that measures the syntactic similarity between parse trees. This kernel computes the inner products of the following feature vector.

$$H(T) = (\lambda^{\frac{size_1}{2}} \# subTree_1(T),..., \lambda^{\frac{size_i}{2}} \# subTree_i(T),..., \lambda^{\frac{size_n}{2}} \# subTree_n(T)) \quad (2)$$
$$0 < \lambda \le 1$$

Each feature of this vector is the occurrence count of a sub-tree type in the parse tree decayed exponentially by the parameter λ. Without this decaying mechanism used to retain the kernel values within a fairly small range, the value of the kernel for identical trees becomes far higher than its value for different trees. Term $size_i$ is defined to be the number of rules or internal nodes of the i^{th} sub-tree type. Samples of such sub-trees are shown in Fig. 1 for a simple parse tree. Since the number of sub-trees of a tree is exponential in its size (Collins and Duffy, 2001), direct inner product calculation is computationally infeasible. Consequently, Collins and Duffy (2001) proposed an ingenious kernel function that implicitly calculates the inner product in $O(N_1 \times N_2)$ time on the trees of size N_1 and N_2.

4 A Generalized Convolution Tree Kernel

In order to describe the kernel, a feature vector over the syntactic parse tree is firstly defined in eq. (3), in which the i^{th} feature equals the weighted sum of the number of instances of sub-tree type i^{th} in the tree.

Function $I_{subtree_i}(n)$ is an indicator function that returns 1 if the $subtree_i$ occurs with its root at node n and 0 otherwise. As described in eq. (4),

function *tw(T)* (which stands for "tree weight") assigns a weight to a tree T which is equal to the product of the weights of all its nodes.

$$H(T) = (\sum_{n\in T}[I_{subtree_1}(n)\times tw(subtree_1(n))],..., \sum_{n\in T}[I_{subtree_i}(n)\times tw(subtree_i(n))],..., \sum_{n\in T}[I_{subtree_m}(n)\times tw(subtree_m(n))]) \quad (3)$$

$$tw(T) = \prod_{n\in InternalNodes(T)} inw(n) \times \prod_{n\in ExternalNodes(T)} enw(n) \quad (4)$$

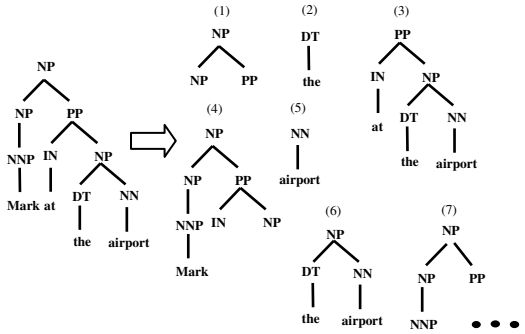

Figure 1. Samples of sub-trees used in convolution tree kernel calculation.

Since each node of the whole syntactic tree can either happen as an internal node or as an external node of a supposed sub-tree (presuming its existence in the sub-tree), two types of weights are respectively associated to each node by the functions $inw(n)$ and $enw(n)$ (which respectively stand for "internal node weight" and "external node weight"). For instance, in Fig. 1, the node with label PP is an external node for sub-trees (1) and (7) while it is an internal node of sub-trees (3) and (4).

$$\begin{aligned} K(T_1,T_2) &= < H(T_1), H(T_2) > \\ &= \sum_i [\ [\sum_{n_1\in T_1} I_{subtree_i}(n_1)\times tw(subTree_i(n_1))] \\ &\quad \times [\sum_{n_2\in T_2} I_{subtree_i}(n_2)\times tw(subTree_i(n_2))]] \quad (5) \\ &= \sum_{n_1\in T_1}\sum_{n_2\in T_2}[\sum_i I_{subtree_i}(n_1)\times I_{subtree_i}(n_2)\times \\ &\quad tw(subTree_i(n_1))\times tw(subTree_i(n_2))] \\ &= \sum_{n_1\in T_1}\sum_{n_2\in T_2} C_{gc}(n_1,n_2) \end{aligned}$$

As shown in eq. (5), A similar procedure to (Collins and Duffy, 2001) can be employed to develop a kernel function for the calculation of dot products on *H(T)* vectors. According to eq. (5) the calculation of the kernel finally leads to the sum of a $C_{gc}(n_1,n_2)$ function over all tree node pairs of T_1 and T_2. Function $C_{gc}(n_1,n_2)$ is the weighted sum of the common sub-trees rooted at n_1 and n_2, and can be recursively computed in a similar way to function $C(n_1,n_2)$ of (Collins and Duffy, 2001) as follows.

(1) if the production rules of nodes n_1 and n_2 are different then $C_{gc}(n_1,n_2) = 0$

(2) else if n_1 and n_2 are the same pre-terminals (the same part of speeches) then

$$C_{gc}(n_1,n_2) = inw(n_1)\times enw(child(n_1))\times inw(n_2)\times enw(child(n_2))$$

(3) else if both n_1 and n_2 have the same production rules then

$$C_{gc}(n_1,n_2) = inw(n_1)\times inw(n_2)\times \prod_i [enw(child_i(n_1))\times enw(child_i(n_2)) + C_{gc}(child_i(n_1), child_i(n_2))]$$

In the first case, when the two nodes represent different production rules they can't accordingly have any sub-trees in common. In the second case, there is exactly one common sub-tree of size two. It should be noted that all the leaf nodes of the tree (or words of the sentence) are considered identical in the calculation of the tree kernel. The value of the function in this case is the weight of this common sub-tree. In the third case, when the nodes generally represent the same production rules the weighted sum of the common sub-trees are calculated recursively. The equation holds because the existence of common sub-trees rooted at n_1 and n_2 implies the existence of common sub-trees rooted at their corresponding children, which can be combined multiplicatively to form their parents' common sub-trees.

Due to the equivalent procedure of kernel calculation, this generalized version of the tree kernel preserves the nice $O(N_1\times N_2)$ time complexity property of the original kernel. It is worthy of note that in (Moschitti, 2006b) a sorting based method is proposed for the fast implementation of such tree kernels that reduces the average running time to $O(N_1+N_2)$.

The generalized kernel can be converted to CD'01 kernel by defining $inw(n) = \sqrt{\lambda}$ and $enw(n) = 1$. Likewise, other definitions can be utilized to produce other useful sub-kernels.

5 Kernels for Relation Extraction

In this section, three sub-kernels of the generalized convolution tree kernel will be proposed for relation extraction. Using the embedded weights of the generalized kernel, these sub-kernels differentiate among sub-trees based on their expected relevance to semantic relations. More specifically, the sub-trees are weighted according to how their nodes interact to the arguments of the relation.

5.1 Argument Ancestor Path Kernel (AAP)

Definition of weighting functions is shown in eq. (6) and (7). Parameter $0 < \alpha \leq 1$ is a decaying parameter similar to λ.

$$inw(n) = \begin{cases} \alpha & \text{if } n \text{ is on the argument ancestor path} \\ & \text{or a direct child of a node on it} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

$$enw(n) = \begin{cases} 1 & \text{if } n \text{ is on the argument ancestor path} \\ & \text{or a direct child of a node on it} \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

This weighting method is equivalent to applying CD'01 tree kernel (by setting $\lambda = \alpha^2$) on a portion of the parse tree that exclusively includes the arguments ancestor nodes and their direct children.

5.2 Argument Ancestor Path Distance Kernel (AAPD)

$$inw(n) = \alpha^{\frac{Min(AAPDist(n,\arg_1), AAPDist(n,\arg_2))}{MAX\_DIST}} \quad (8)$$

$$enw(n) = \alpha^{\frac{Min(AAPDist(n,\arg_1), AAPDist(n,\arg_2))}{MAX\_DIST}} \quad (9)$$

Definition of weighting functions is shown in eq. (8) and (9). Both functions have identical definitions for this kernel.

Function *AAPDist(n,arg)* calculates the distance of the node *n* from the argument *arg* on the parse tree as illustrated by Fig. 2. MAX_DIST is used for normalization, and is the maximum of *AAPDist(n,arg)* in the whole tree. In this way, the closer a tree node is to one of the arguments ancestor path, the less it is decayed by this weighting method.

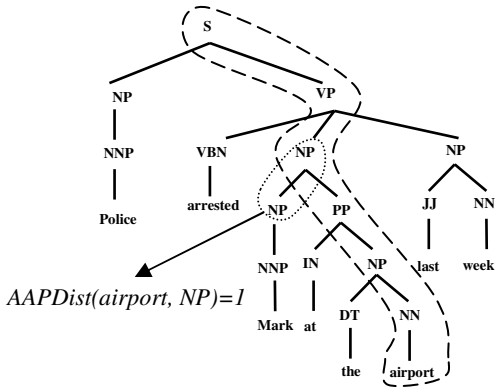

Figure 2. The syntactic parse tree of the sentence "Police arrested Mark at the airport last week" that conveys a Phys.Located(Mark, airport) relation. The ancestor path of the argument "airport" (dashed curve) and the distance of the node NP of "Mark" from it (dotted curve) is shown.

5.3 Threshold Sensitive Argument Ancestor Path Distance Kernel (TSAAPD)

This kernel is intuitively similar to the previous kernel but uses a rough threshold based decaying technique instead of a smooth one. The definition of weighting functions is shown in eq. (10) and (11). Both functions are again identical in this case.

$$inw(n) = \begin{cases} 1 & AAPDist(n) \leq Threshold \\ \alpha & AAPDist(n) \geq Threshold \end{cases} \quad (10)$$

$$enw(n) = \begin{cases} 1 & AAPDist(n) \leq Threshold \\ \alpha & AAPDist(n) \geq Threshold \end{cases} \quad (11)$$

6 Experiments

6.1 Experiments Setting

The proposed kernels are evaluated on ACE-2005 multilingual corpus (Walker et al., 2006). In order to avoid parsing problems, the more formal parts of the corpus in "news wire" and "broadcast news" sections are used for evaluation as in (Zhang et al., 2006b).

Relation / Kernel	PER-SOC	ART	GEN-AFF	ORG-AFF	PART-WHOLE	PHYS
CD'01	0.62	**0.51**	0.09	0.43	**0.30**	0.32
AAP	0.58	0.49	0.10	0.43	0.28	**0.36**
AAPD	0.70	0.50	**0.12**	0.43	0.29	0.29
TSAAPD-0	0.63	0.48	0.11	0.43	**0.30**	0.33
TSAAPD-1	**0.73**	0.47	0.11	**0.45**	0.28	0.33

Table 1: The F_1-Measure value is shown for every kernel on each ACE-2005 main relation type. For every relation type the best result is shown in bold font.

We have used LIBSVM (Chang and Lin 2001) java source for the SVM classification and Stanford NLP package[1] for tokenization, sentence segmentation and parsing.

Following [Bunescu and Mooney, 2007], every pair of entities within a sentence is regarded as a negative relation instance unless it is annotated as a positive relation in the corpus. The total number of negative training instances, constructed in this way, is about 20 times more than the number of annotated positive instances. Thus, we also imposed the restriction of maximum argument distance of 10 words. This constraint eliminates half of the negative constructed instances while slightly decreases positive instances. Nevertheless, since the resulted training set is still unbalanced, we used LIBSVM weighting mechanism. Precisely, if there are P positive and N negative instances in the training set, a weight value of N/P is used for positive instances while the default weight value of 1 is used for negative ones.

A binary SVM is trained for every relation type separately, and type compatible annotated and constructed relation instances are used to train it. For each relation type, only type compatible relation instances are exploited for training. For example to learn an ORG-AFF relation (which applies to (PER, ORG) or (ORG, ORG) argument types) it is meaningless to use a relation instance between two entities of type PERSON. Moreover, the total number of training instances used for training every relation type is restricted to 5000 instances to shorten the duration of the evaluation process. The reported results are achieved using a 5-fold cross validation method.

The kernels AAP, AAPD and TSAAPD-0 (TSAAPD with threshold = 0) and TSAAPD-1 (TSAAPD with threshold = 1) are compared with CD'01 convolution tree kernel. All the kernels except for AAP are computed on the PT portion described in section 2. AAP is computed over the MCT tree portion which is also proposed by (Zhang et al., 2006a) and is the sub-tree rooted at the first common ancestor of relation arguments.

For the proposed kernels α is set to 0.44 which is tuned on a development set that contained 5000 instances of type PHYS. The λ parameter of CD'01 kernel is set to 0.4 according to (Zhang et al., 2006a). The C parameter of SVM classification is set to 2.4 for all the kernels after tuning it individually for each kernel on the mentioned development set.

6.2 Experiments Results

The results of the experiments are shown in Table 1. The proposed kernels outperform the original CD'01 kernel in four of the six relation types. The performance of TSAAPD-1 is especially remarkable because it is the best kernel in ORG-AFF and PER-SOC relations. It particularly performs very well in the extraction of PER-SOC relation with an F_1-measure of 0.73. It should be noted that the general low performance of all the kernels on the GEN-AFF type is because of its extremely small number of annotated instances in the training set (40 in 5000). The AAPD kernel has the best performance with a remarkable improvement over the Collins kernel in GEN-AFF relation type.

The results clearly demonstrate that the nodes closer to the ancestor path of relation arguments contain the most useful syntactic features for relation extraction

7 Conclusion

In this paper, we proposed a generalized convolution tree kernel that can generate various syntactic sub-kernels including the CD'01 kernel.

1 http://nlp.stanford.edu/software/index.shtml

The kernel is generalized by assigning weights to the sub-trees. The weight of a sub-tree is the product of the weights assigned to its nodes by two types of weighting functions. In this way, impacts of the tree nodes on the kernel value can be discriminated purposely based on the application. Context information can also be injected to the kernel via context sensitive weighting mechanisms.

Using the generalized kernel, various sub-kernels can be produced by different definitions of the two weighting functions. We consequently used the generalized kernel for systematic generation of useful kernels in relation extraction. In these kernels, the closer a node is to the relation arguments ancestor paths, the less it is decayed by the weighting functions. Evaluation on the ACE-2005 main relation types demonstrates the effectiveness of the proposed kernels. They show remarkable performance improvement over CD'01 kernel.

8 Future Work

Although the path-enclosed tree portion (PT) (Zhang et al., 2006a) seems to be an appropriate portion of the syntactic tree for relation extraction, it only takes into account the syntactic information between the relation arguments, and discards many useful features (before and after the arguments features). It seems that the generalized kernel can be used with larger tree portions that contain syntactic features before and after the arguments, because it can be more easily targeted to related features.

Currently, the proposed weighting mechanisms are solely based on the location of the tree nodes in the parse tree; however other useful information such as labels of nodes can also be used in weighting.

Another future work can be utilizing the generalized kernel for other applicable NLP tasks such as co-reference resolution.

Acknowledgement

This work is supported by Iran Telecommunication Research Centre under contract No. 500-7725.

References

Boser B. E., Guyon I., and Vapnik V. 1992. *A training algorithm for optimal margin classifiers*. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pages 144-152. ACM Press.

Bunescu R. C. and Mooney R. J. 2005a. *A Shortest Path Dependency Kernel for Relation Extraction*. EMNLP-2005

Bunescu R. C. and Mooney R. J. 2005b. *Subsequence kernels for relation extraction*. NIPS-2005.

Bunescu R. C. and Mooney R. J. 2007. *Learning for Information Extraction: From Named Entity Recognition and Disambiguation to Relation Extraction, Ph.D. Thesis*. Department of Computer Sciences, University of Texas at Austin.

Chang, C.-C. and C.-J. Lin 2001. *LIBSVM: a library for support vector machines*. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Collins M. and Duffy N. 2001. *Convolution Kernels for Natural Language*. NIPS-2001

Cortes C. and Vapnik V. 1995. *Support-vector network*. Machine Learning. 20, 273-297.

Li J., Zhang Z., Li X. and Chen H. 2008. *Kernel-based learning for biomedical relation extraction*. J. Am. Soc. Inf. Sci. Technol. 59, 5, 756–769.

Moschitti A. 2006a. *Making tree kernels practical for natural language learning*. EACL-2006.

Moschitti A. 2006b. *Syntactic kernels for natural language learning: the semantic role labeling case*. HLT-NAACL-2006 (short paper)

Walker, C., Strassel, S., Medero J. and Maeda, K. 2006. *ACE 2005 Multilingual Training Corpus.* Linguistic Data Consortium, Philadelphia.

Zhang M., Zhang J. and SU j. 2006a. *Exploring syntactic features for relation extraction using a convolution tree kernel. HLT-NAACL-2006*.

Zhang M., Zhang J., Su J. and Zhou G.D. 2006b. *A Composite Kernel to Extract Relations between Entities with both Flat and Structured* COLINGACL-2006: 825-832.

Zhou G.D., Su J, Zhang J. and Zhang M. 2005. *Exploring Various Knowledge in Relation Extraction*. ACL-2005

Zhou G.D., Zhang M., Ji D.H. and Zhu Q.M. 2007. *Tree Kernel-based Relation Extraction with Context-Sensitive Structured Parse Tree Information.* EMNLP-CoNLL-2007

Towards Building a Competitive Opinion Summarization System: Challenges and Keys

Elena Lloret*, Alexandra Balahur, Manuel Palomar and Andrés Montoyo
Department of Software and Computing Systems
University of Alicante
Apartado de Correos 99, E-03080, Alicante, Spain
{elloret, abalahur, mpalomar, montoyo}@dlsi.ua.es

Abstract

This paper presents an overview of our participation in the TAC 2008 Opinion Pilot Summarization task, as well as the proposed and evaluated post-competition improvements. We first describe our opinion summarization system and the results obtained. Further on, we identify the system's weak points and suggest several improvements, focused both on information content, as well as linguistic and readability aspects. We obtain encouraging results, especially as far as F-measure is concerned, outperforming the competition results by approximately 80%.

1 Introduction

The Opinion Summarization Pilot (OSP) task within the TAC 2008 competition consisted in generating summaries from answers to opinion questions retrieved from blogs (the Blog06[1] collection). The questions were organized around 25 targets – persons, events, organizations etc. Additionally, a set of text snippets that contained the answers to the questions were provided by the organizers, their use being optional. An example of target, question and provided snippet is given in Figure 1.

Target : George Clooney
Question: Why do people like George Clooney?
Snippet 1: 1050 BLOG06-20060125-015-0025581509 he is a great actor

Figure 1. *Examples of target, question and snippet*

The techniques employed by the participants were mainly based on the already existing summarization systems. While most participants added new features (sentiment, pos/neg sentiment, pos/neg opinion) to account for the presence of positive opinions or negative ones - CLASSY (Conroy and Schlessinger, 2008); CCNU (He et al.,2008); LIPN (Bossard et al., 2008); IIITSum08 (Varma et al., 2008) -, efficient methods were proposed focusing on the retrieval and filtering stage, based on polarity – DLSIUAES (Balahur et al., 2008) - or on separating information rich clauses - italica (Cruz et al., 2008). In general, previous work in opinion mining includes document level sentiment classification using supervised (Chaovalit and Zhou, 2005) and unsupervised methods (Turney, 2002), machine learning techniques and sentiment classification considering rating scales (Pang, Lee and Vaithyanathan, 2002), and scoring of features (Dave, Lawrence and Pennock, 2003). Other research has been conducted in analysing sentiment at a sentence level using bootstrapping techniques (Riloff and Wiebe, 2003), finding strength of opinions (Wilson, Wiebe and Hwa, 2004), summing up orientations of opinion words in a sentence (Kim and Hovy, 2004), and identifying opinion holders (Stoyanov and Cardie, 2006). Finally, fine grained, feature-based opinion summarization is defined in (Hu and Liu, 2004).

2 Opinion Summarization System

In order to tackle the OSP task, we considered the use of two different methods for opinion mining and summarization, differing mainly with respect to the use of the optional text snippets provided. Our first approach (the Snippet-driven Approach)

*Elena Lloret is funded by the FPI program (BES-2007-16268) from the Spanish Ministry of Science and Innovation, under the project TEXT-MESS (TIN-2006-15265)
[1]http://ir.dcs.gla.ac.uk/test_collections/access_to_data.html

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 72–77,
Boulder, Colorado, June 2009.

used these snippets, whereas the second one (Blog-driven Approach) found the answers directly in the corresponding blogs. A general overview of the system's architecture is shown in Figure 2, where three main parts can be distinguished: the question processing stage, the snippets processing stage (only carried out for the first approach), and the final summary generation module. Next, the main steps involved in each process will be explained in more detail.

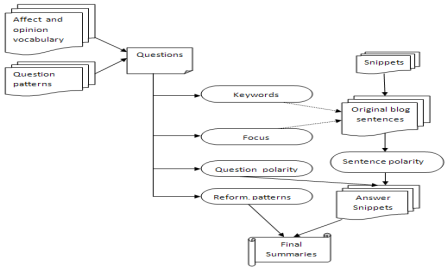

Figure 2. *System architecture*

The first step was to determine the polarity of each question, extract the keywords from each of them and finally, build some patterns of reformulation. The latter were defined in order to give the final summary an abstract nature, rather than a simple joining of sentences. The polarity of the question was determined using a set of created patterns, whose goal was to extract for further classification the nouns, verbs, adverbs or adjectives indicating some kind of polarity (positive or negative). These extracted words, together with their determiners, were classified using the emotions lists in WordNet Affect (Strapparava and Valitutti, 2005), jointly with the emotions lists of attitudes, triggers resource (Balahur and Montoyo, 2008 [1]), four created lists of attitudes, expressing criticism, support, admiration and rejection and two categories for value (good and bad), taking for the opinion mining systems in (Balahur and Montoyo, 2008 [2]). Moreover, the focus of each question was automatically extracted using the Freeling[2] Named Entity Recognizer module. This information was used to determine whether or not all the questions within the same topic had the same focus, as well as be able to decide later on which text snippet belonged to which question.

Regarding the given text snippets, we also computed their polarity and their focus. The polarity was calculated as a vector similarity between the snippets and vectors constructed from the list of sentences contained in the ISEAR corpus (Scherer and Wallbot, 1997), WordNet Affect emotion lists of anger, sadness, disgust and joy and the emotion triggers resource, using Pedersen's Text Similarity Package.[3]

Concerning the blogs, our opinion mining and summarization system is focused only on plain text; therefore, as pre processing stage, we removed all unnecessary tags and irrelevant information, such as links, images etc. Further on, we split the remaining text into individual sentences. A matching between blogs' sentences and text snippets was performed so that a preliminary set of potential meaningful sentences was recorded for further processing. To achieve this, snippets not literally contained in the blogs were tokenized and stemmed using Porter's Stemmer,[4] and stop words were removed in order to find the most similar possible sentence associated with it. Subsequently, by means of the same Pedersen Text Similarity Package as for computing the snippets' polarity, we computed the similarity between the given snippets and this created set of potential sentences. We extracted the complete blog sentences to which each snippet was related. Further on, we extracted the focus for each blog phrase sentence as well. Then, we filtered redundant sentences using a naïve similarity based approach. Once we obtained the possible answers, we used Minipar[5] to filter out incomplete sentences.

Having computed the polarity for the questions and snippets, and set out the final set of sentences to produce the summary, we bound each sentence to its corresponding question, and we grouped all sentences which were related to the same question together, so that we could generate the language for this group, according to the patterns of reformulation previously mentioned. Finally, the speech style was changed to an impersonal one, in order to avoid directly expressed opinion sentences. A POS-tagger tool (TreeTagger[6]) was used to identify third person verbs and change them to a neutral style. A set of rules to identify

[2] http://garraf.epsevg.upc.es/freeling/

[3] http://www.d.umn.edu/~tpederse/text-similarity.html

[4] http://tartarus.org/~martin/PorterStemmer/

[5] http://www.cs.ualberta.ca/~lindek/minipar.htm

[6] http://www.ims.uni-tuttgart.de/projekte/corplex/TreeTagger/

pronouns was created, and they were also changed to the more general pronoun "they" and its corresponding forms, to avoid personal opinions.

3 Evaluation

Table 1 shows the final results obtained by our approaches in the TAC 2008 Opinion Pilot (the rank among the 36 participating systems is shown in brackets for each evaluation measure). Both of our approaches were totally automatic, and the only difference between them was the use of the given snippets in the first one (A1) and not in the second (A2). The column numbers stand for the following average scores: summarizerID (1); pyramid F-score (Beta=1) (2), grammaticality (3); non-redundancy (4); structure/coherence (including focus and referential clarity) (5); overall fluency/readability (6); overall responsiveness (7).

1	2	3	4	5	6	7
A1	0.357 (7)	4.727 (8)	5.364 (28)	3.409 (4)	3.636 (16)	5.045 (5)
A2	0.155 (23)	3.545 (36)	4.364 (36)	3.091 (13)	2.636 (36)	2.227 (28)

Table 1. *Evaluation results*

As it can be noticed from Table 1, our system performed well regarding F-measure, the first run being classified 7th among the 36 evaluated. As far as the structure and coherence are concerned, the results were also good, placing the first approach in the fourth. Also worth mentioning is the good performance obtained regarding the overall responsiveness, where A1 ranked 5th. Generally speaking, the results for A1 showed well-balanced among all the criteria evaluated, except for non redundancy and grammaticality. For the second approach, results were not as good, due to the difficulty in selecting the appropriate opinion blog sentence by only taking into account the keywords of the question.

4 Post-competition tests, experiments and improvements

When an exhaustive examination of the nuggets used for evaluating the summaries was done, we found some problems that are worth mentioning.

a) Some nuggets with high score did not exist in the snippet list (e.g. "*When buying from CARMAX, got a better than blue book trade-in on old car"* (0.9)).
b) Some nuggets for the same target express the same idea, despite their not being identical (e.g. "*NAFTA needs to be renegotiated to protect Canadian sovereignty"* and "*Green Party: Renegotiate NAFTA to protect Canadian Sovereignty"*).
c) The meaning of one nugget can be deduced from another's (e.g. "*reasonably healthy food" and "sandwiches are healthy"*).
d) Some nuggets are not very clear in meaning (e.g. "*hot", "fun").*
e) A snippet can be covered by several nuggets (e.g. both nuggets "*it is an honest book"* and *"it is a great book"* correspond to the same snippet *"It was such a great book- honest and hard to read (content not language difficulty)").*

On the other hand, regarding the use of the optional snippets, the main problem to address is to remove redundancy, because many of them are repeated for the same target, and we have to determine which snippet represents better the idea for the final summary, in order to avoid noisy irrelevant information.

4.1 Measuring the Performance of a Generic Summarization System

Several participants in the TAC 2008 edition performed the OSP task by using generic summarization systems. Most were adjusted by integrating an opinion classifier module so that the task could be fulfilled, but some were not (Bossard et al., 2008), (Hendrickx and Bosma, 2008). This fact made us realize that a generic summarizer could be used to achieve this task. We wanted to analyze the effects of such a kind of summarizer to produce opinion summaries. We followed the approach described in (Lloret et al., 2008). The main idea employed is to score sentences of a document with regard to the word frequency count (WF), which can be combined with a Textual Entailment (TE) module.

Although the first approach suggested for opinion summarization obtained much better results in the evaluation than the second one (see Section 3.1), we decided to run the generic system over both approaches, with and without applying TE, to

provide a more extent analysis and conclusions. After preprocessing the blogs and having all the possible candidate sentences grouped together, we considered these as the input for the generic summarizer. The goal of these experiments was to determine whether the techniques used for a generic summarizer would have a positive influence in selecting the main relevant information to become part of the final summary.

4.2 Results and Discussion

We re-evaluated the summaries generated by the generic system following the nuggets' list provided by the TAC 2008 organization, and counting manually the number of nuggets that were covered in the summaries. This was a tedious task, but it could not be automatically performed because of the fact that many of the provided nuggets were not found in the original blog collection. After the manual matching of nuggets and sentences, we computed the average Recall, Precision and F-measure (Beta =1) in the same way as in the TAC 2008 was done, according to the number and weight of the nuggets that were also covered in the summary. Each nugget had a weight ranging from 0 to 1 reflecting its importance, and it was counted only once, even though the information was repeated within the summary.

The average for each value was calculated taking into account the results for all the summaries in each approach. Unfortunately, we could not measure criteria such as readability or coherence as they were manually evaluated by human experts.

Table 2 points out the results for all the approaches reported. We have also considered the results derived from our participation in the TAC 2008 conference (OpSum-1 and OpSum-2), in order to analyze whether they have been improved or not. From these results it can be stated that the TE module in conjunction with the WF counts, have been very appropriate in selecting the most important information of a document. Although it can be thought that applying TE can remove some meaningful sentences which contained important information, results show the opposite. It benefits the Precision value, because a shorter summary contains greater ratio of relevant information. On the other hand, taking into consideration the F-measure value only, it can be seen that the approach combining TE and WF, for the sentences in the first approach, has beaten significantly the best F-measure result among the participants of TAC 2008 (please see Table 3), increasing its performance by 20% (with respect to WF only), and improving by approximately 80% with respect to our first approach submitted to TAC 2008.

However, a simple generic summarization system like the one we have used here is not enough to produce opinion oriented summaries, since semantic coherence given by the grouping of positive and negative opinions is not taken into account. Therefore, the opinion classification stage must be added in the same manner as used in the competition.

SYSTEM	RECALL	PRECISION	F-MEASURE
OpSum-1	0.592	0.272	0.357
OpSum-2	0.251	0.141	0.155
WF-1	**0.705**	0.392	0.486
TE+WF -1	0.684	**0.630**	**0.639**
WF -2	0.322	0.234	0.241
TE+WF-2	0.292	0.282	0.262

Table 2. *Comparison of the results*

4.3 Improving the quality of summaries

In the evaluation performed by the TAC organization, a manual quality evaluation was also carried out. In this evaluation the important aspects were grammaticality, non-redundancy, structure and coherence, readability, and overall responsiveness. Although our participating systems obtained good F-measure values, in other scores, especially in grammaticality and non-redundancy, the results achieved were very low. Focusing all our efforts in improving the first approach, OpSum-1, non-redundancy and grammaticality verification had to be performed. In this approach, we wanted to test how much of the redundant information would be possible to remove by using a Textual Entailment system similar to (Iftene and Balahur-Dobrescu, 2007), without it affecting the quality of the remaining data. As input for the TE system, we considered the snippets retrieved from the original blog posts. We applied the entailment verification on each of the possible pairs, taking in turn all snippets as Text and Hypothesis with all other snippets as Hypothesis and Text, respectively. Thus, as output, we obtained the list of snippets from which we eliminated those that

are entailed by any of the other snippets. We further eliminated those snippets which had a high entailment score with any of the remaining snippets.

SYSTEM	F-MEASURE
Best system	0.534
Second best system	0.490
OpSum-1 + TE	0.530
OpSum-1	0.357

Table 3. *F-measure results after improving the system*

Table 3 shows that applying TE before generating the final summary leads to very good results increasing the F-measure by 48.50% with respect to the original first approach. Moreover, it can be seen form Table 3 that our improved approach would have ranked in the second place among all the participants, regarding F-measure. The main problem with this approach is the long processing time. We can apply Textual Entailment in the manner described within the generic summarization system presented, successively testing the relation as Snippet1 entails Snippet2?, Snippet1+Snippet2 entails Snippet3? and so on. The problem then becomes the fact that this approach is random, since different snippets come from different sources, so there is no order among them. Further on, we have seen that many problems arise from the fact that extracting information from blogs introduces a lot of noise. In many cases, we had examples such as:

At 4:00 PM John said Starbucks coffee tastes great

John said Starbucks coffee tastes great, always get one when reading New York Times.

To the final summary, the important information that should be added is *"Starbucks coffee tastes great"*. Our TE system contains a rule specifying that the existence or not of a Named Entity in the hypothesis and its not being mentioned in the text leads to the decision of "NO" entailment. For the example given, both snippets are maintained, although they contain the same data.

Another issue to be addressed is the extra information contained in final summaries that is not scored as nugget. As we have seen from our data, much of this information is also valid and correctly answers the questions. Therefore, what methods can be employed to give more weight to some and penalize others automatically?

Regarding the grammaticality criteria, once we had a summary generated we used the module Language Tool[7] as a post-processing step. The errors that we needed correcting included the number matching between nouns and determiners as well as among subject and predicate, upper case for sentence start, repeated words or punctuation marks and lack of punctuation marks. The rules present in the module and that we "switched off", due to the fact that they produced more errors, were those concerning the limit in the number of consecutive nouns and the need for an article before a noun (since it always seemed to want to correct *"Vista"* for *"the Vista"* a.o.). We evaluated by observing the mistakes that the texts contained, and counting the number of remaining or introduced errors in the output. The results obtained can be seen in Table 4.

Problem	Rightly corrected	Wrongly corrected
Match S-P	90%	10%
Noun-det	75%	25%
Upper case	80%	20%
Repeated words	100%	0%
Repeated "."	80%	20%
Spelling mistakes	60%	40%
Unpaired ""/()	100%	0%

Table 4. *Grammaticality analysis*

The greatest problem encountered was the fact that bigrams are not detected and agreement is not made in cases in which the noun does not appear exactly after the determiner. All in all, using this module, the grammaticality of our texts was greatly improved.

5 Conclusions and future work

The Opinion Pilot in the TAC 2008 competition was a difficult task, involving the development of systems including components for QA, IR, polarity classification and summarization. Our contribution presented in this paper resides in proposing an opinion mining and summarization method using different approaches and resources, evaluating each of them in turn. We have shown that using a generic summarization system, we obtain 80% improvement over the results obtained in the competition, with coherence being maintained by using the same polarity classification mechanisms.

[7] http://community.languagetool.org/

Using redundancy removal with TE, as opposed to our initial polarity strength based sentence filtering improved the system performance by almost 50%.
Finally, we showed that grammaticality can be checked and improved using an independent solution given by Language Tool.
Further work includes the improvement of the polarity classification component by using machine learning over annotated corpora and other techniques, such as anaphora resolution. As we could see, the well functioning of this component ensures logic, structure and coherence to the produced summaries. Moreover, we plan to study the manner in which opinion sentences of blogs/bloggers can be coherently combined.

References

Balahur, A., Lloret, E., Ferrández, Ó., Montoyo, A., Palomar, M., Muñoz, R., The DLSIUAES Team's Participation in the TAC 2008 Tracks. In Proceedings of the Text Analysis Conference (TAC), 2008.

Balahur, A. and Montoyo, A. [1]. An Incremental Multilingual Approach to Forming a Culture Dependent Emotion Triggers Database. In Proceedings of the 8th International Conference on Terminology and Knowledge Engineering, 2008.

Balahur, A. and Montoyo, A. [2]. Multilingual Feature--driven Opinion Mining and Summarization from Customer Reviews. In Lecture Notes in Computer Science 5039, pg. 345-346.

Bossard, A., Généreux, M. and Poibeau, T.. Description of the LIPN systems at TAC 2008: Summarizing information and opinions. In Proceedings of the Text Analysis Conference (TAC), 2008.

Chaovalit, P., Zhou, L. 2005. Movie Review Mining: a Comparison between Supervised and Unsupervised Classification Approaches. In Proceedings of HICSS-05, the 38th Hawaii International Conference on System Sciences.

Cruz, F., Troyani, J.A., Ortega, J., Enríquez, F. The Italica System at TAC 2008 Opinion Summarization Task. In Proceedings of the Text Analysis Conference (TAC), 2008.

Cui, H., Mittal, V., Datar, M. 2006. Comparative Experiments on Sentiment Classification for Online Product Reviews. In Proceedings of the 21st National Conference on Artificial Intelligence AAAI 2006.

Dave, K., Lawrence, S., Pennock, D. 2003. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In Proceedings of WWW-03.

Lloret, E., Ferrández, O., Muñoz, R. and Palomar, M. A Text Summarization Approach under the Influence of Textual Entailment. In Proceedings of the 5th International Workshop on Natural Language Processing and Cognitive Science (NLPCS 2008), pages 22–31, 2008.

Gamon, M., Aue, S., Corston-Oliver, S., Ringger, E. 2005. Mining Customer Opinions from Free Text. Lecture Notes in Computer Science.

He, T., Chen, J., Gui, Z., Li, F. CCNU at TAC 2008: Proceeding on Using Semantic Method for Automated Summarization Yield. In Proceedings of the Text Analysis Conference (TAC), 2008.

Hendrickx, I. and Bosma, W.. Using coreference links and sentence compression in graph-based summarization. In Proceedings of the Text Analysis Conference (TAC), 2008.

Hu, M., Liu, B. 2004. Mining Opinion Features in Customer Reviews. In Proceedings of 19th National Conference on Artificial Intelligence AAAI.

Iftene, A., Balahur-Dobrescu, A. Hypothesis Transformation and Semantic Variability Rules for Recognizing Textual Entailment. In Proceedings of the ACL 2007 Workshop on Textual Entailment and Paraphrasis, 2007.

Kim, S.M., Hovy, E. 2004. Determining the Sentiment of Opinions. In Proceedings of COLING 2004.

Pang, B., Lee, L., Vaithyanathan, S. 2002. Thumbs up? Sentiment classification using machine learning techniques. In Proceedings of EMNLP-02, the Conference on Empirical Methods in Natural Language Processing.

Riloff, E., Wiebe, J. 2003 Learning Extraction Patterns for Subjective Expressions. In Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing.

Scherer, K. and Wallbott, H.G. The ISEAR Questionnaire and Codebook, 1997.

Stoyanov, V., Cardie, C. 2006. Toward Opinion Summarization: Linking the Sources. In: COLING-ACL 2006 Workshop on Sentiment and Subjectivity in Text.

Strapparava, C. and Valitutti, A. "WordNet-Affect: an affective extension of WordNet". In Proceedings ofthe 4th International Conference on Language Resources and Evaluation, 2004, pp. 1083-1086.

Turney, P., 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In Proceedings of the 40th Annual Meeting of the ACL

Varma, V., Pingali, P., Katragadda, R., Krisha, S., Ganesh, S., Sarvabhotla, K., Garapati, H., Gopisetty, H.,, Reddy, V.B., Bysani, P., Bharadwaj, R. IIT Hyderabad at TAC 2008. In Proceedings of the Text Analysis Conference (TAC), 2008.

Wilson, T., Wiebe, J., Hwa, R. 2004. Just how mad are you? Finding strong and weak opinion clauses. In: Proceedings of AAAI 2004.

Domain-Independent Shallow Sentence Ordering

Thade Nahnsen
School of Informatics
University of Edinburgh
T.Nahnsen@sms.ed.ac.uk

Abstract

We present a shallow approach to the sentence ordering problem. The employed features are based on discourse entities, shallow syntactic analysis, and temporal precedence relations retrieved from VerbOcean. We show that these relatively simple features perform well in a machine learning algorithm on datasets containing sequences of events, and that the resulting models achieve optimal performance with small amounts of training data. The model does not yet perform well on datasets describing the consequences of events, such as the destructions after an earthquake.

1 Introduction

Sentence ordering is a problem in many natural language processing tasks. While it has, historically, mainly been considered a challenging problem in (concept-to-text) language generation tasks, more recently, the issue has also generated interest within summarization research (Barzilay, 2003; Ji and Pulman, 2006). In the spirit of the latter, this paper investigates the following questions: (1) Does the topic of the text influence the factors that are important to sentence ordering? (2) Which factors are most important for determining coherent sentence orderings? (3) How much performance is gained when using deeper knowledge resources?

Past research has investigated a wide range of aspects pertaining to the ordering of sentences in text. The most prominent approaches include: (1) temporal ordering in terms of publication date (Barzilay, 2003), (2) temporal ordering in terms of textual cues in sentences (Bollegala et al., 2006), (3) the topic of the sentences (Barzilay, 2003), (4) coherence theories (Barzilay and Lapata, 2008), e.g., Centering Theory, (5) content models (Barzilay and Lee, 2004), and (6) ordering(s) in the underlying documents in the case of summarisation (Bollegala et al., 2006; Barzilay, 2003).

2 The Model

We view coherence assessment, which we recast as a sentence ordering problem, as a machine learning problem using the feature representation discussed in Section 2.1. It can be viewed as a ranking task because a text can only be more or less coherent than some other text. The sentence ordering task used in this paper can easily be transformed into a ranking problem. Hence, paralleling Barzilay and Lapata (2008), our model has the following structure.

The data consists of alternative orderings (x_{ij}, x_{ik}) of the sentences of the same document d_i. In the training data, the preference ranking of the alternative orderings is known. As a result, training consists of determining a parameter vector $\mathbf{w}$ that minimizes the number of violations of pairwise rankings in the training set, a problem which can be solved using SVM constraint optimization (Joachims, 2002). The following section explores the features available for this optimization.

2.1 Features

Approaches to sentence ordering can generally be categorized as knowledge-rich or knowledge-lean. Knowledge-rich approaches rely on manually created representations of sentence orderings using do-

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 78–83,
Boulder, Colorado, June 2009.

main communication knowledge.

Barzilay and Lee (2004)'s knowledge-lean approach attempts to automate the inference of knowledge-rich information using a distributional view of content. In essence, they infer a number of topics using clustering. The clusters are represented by corresponding states in a hidden Markov model, which is used to model the transitions between topics.

Lapata (2003), in contrast, does not attempt to model topics explicitly. Instead, she reduces sentence ordering to the task of predicting the next sentence given the previous sentence, which represents a coarse attempt at capturing local coherence constraints. The features she uses are derived from three categories - verbs, nouns, and dependencies - all of which are lexicalised. Her system thereby, to some extent, learns a precedence between the words in the sentences, which in turn represent topics.

Ji and Pulman (2006) base their ordering strategy not only on the directly preceding sentence, but on all preceding sentences. In this way, they are able to avoid a possible topic bias when summarizing multiple documents. This is specific to their approach as both Lapata (2003)'s and Barzilay and Lee (2004)'s approaches are not tailored to summarization and therefore do not experience the topic bias problem.

The present paper deviates from Lapata (2003) insofar as we do not attempt to learn the ordering preferences between pairs of sentences. Instead, we learn the ranking of documents. The advantage of this approach is that it allows us to straightforwardly discern the individual value of various features (cf. Barzilay and Lapata (2008)).

The methods used in this paper are mostly shallow with the exception of two aspects. First, some of the measures make use of WordNet relations (Fellbaum, 1998), and second, some use the temporal ordering provided by the "happens-before" relation in VerbOcean (Chklovski and Pantel, 2004). While the use of WordNet is self-explanatory, its effect on sentence ordering algorithms does not seem to have been explored in any depth. The use of VerbOcean is meant to reveal the degree to which common sense orderings of events affect the ordering of sentences, or whether the order is reversed.

With this background, the sentence ordering features used in this paper can be grouped into three categories:

2.1.1 Group Similarity

The features in this category are inspired by discourse entity-based accounts of local coherence. Yet, in contrast to Barzilay and Lapata (2008), who employ the syntactic properties of the respective occurrences, we reduce the accounts to whether or not the entities occur in subsequent sentences (similar to Karamanis (2004)'s NOCB metric). We also investigate whether using only the information from the head of the noun group (cf. Barzilay and Lapata (2008)) suffices, or whether performance is gained when allowing the whole noun group in order to determine similarity. Moreover, as indicated above, some of the noun group measures make use of WordNet synonym, hypernym, hyponym, antonym relationships. For completeness, we also consider the effects of using verb groups and whole sentences as syntactic units of choice.

2.1.2 Temporal Ordering

This set of features uses information on the temporal ordering of sentences, although it currently only includes the "happens-before" relations in VerbOcean.

2.1.3 Longer Range Relations

The group similarity features only capture the relation between a sentence and its immediate successor. However, the coherence of a text is clearly not only defined by direct relations, but also requires longer range relations between sentences (e.g., Barzilay and Lapata (2008)). The features in this section explore the impact of such relations on the coherence of the overall document as well as the appropriate way of modeling them.

3 Experiments

This section introduces the datasets used for the experiments, describes the experiments, and discusses our main findings.

3.1 Evaluation Datasets

The three datasets used for the automatic evaluation in this paper are based on human-generated texts (Table 1). The first two are the earthquake and accident datasets used by Barzilay and Lapata (2008).

Each of these sets consists of 100 datasets in the training and test sets, respectively, as well as 20 random permutations for each text.

The third dataset is similar to the first two in that it contains original texts and random permutations. In contrast to the other two sources, however, this dataset is based on the human summaries from DUC 2005 (Dang, 2005). It comprises 300 human summaries on 50 document sets, resulting in a total of 6,000 pairwise rankings split into training and test sets. The source furthermore differs from Barzilay and Lapata (2008)'s datasets in that the content of each text is not based on one individual event (an earthquake or accident), but on more complex topics followed over a period of time (e.g., the espionage case between GM and VW along with the various actions taken to resolve it). Since the different document sets cover completely different topics the third dataset will mainly be used to evaluate the topic-independent properties of our model.

Dataset	*Training*	*Testing*
Earthquakes	1,896	2,056
Accidents	2,095	2,087
DUC2005	up to 3,300	2,700

Table 1: Number of pairwise rankings in the training and test sets for the three datasets

3.2 Experiment 1

In the first part of this experiment, we consider the problem of the granularity of the syntactic units to be used. That is, does it make a difference whether we use the words in the sentence, the words in the noun groups, the words in the verb groups, or the words in the respective heads of the groups to determine coherence? (The units are obtained by processing the documents using the LT-TTT2 tools (Grover and Tobin, 2006); the lemmatizer used by LT-TTT2 is morpha (Minnen and Pearce, 2000).) We also consider whether lemmatization is beneficial in each of the granularities.

The results - presented in Table 2 - indicate that considering only the heads of the verb and noun groups separately provides the best performance. In particular, the heads outperform the whole groups, and the heads separately also outperform noun and verb group heads together. As for the question of whether lemmatization provides better results, one needs to distinguish the case of noun and verb groups. For noun groups, lemmatization improves performance, which can mostly be attributed to singular and plural forms. In the case of verb groups, however, the lemmatized version yields worse results than the surface forms, a fact mainly explained by the tense and modality properties of verbs.

Syntactic Unit	*Processing*	*Accuracy*	
		Acc	Earth
sentence	surface form	52.27	14.21
	lemma	52.27	12.04
heads sentence	surface form	77.35	60.30
	lemma	73.18	61.67
noun group	surface form	80.14	59.84
	lemma	81.58	59.54
head NG	surface form	80.49	59.75
	lemma	81.65	59.12
verb group	surface form	71.57	68.14
	lemma	53.40	68.01
head VG	surface form	71.15	68.39
	lemma	53.76	67.85

Table 2: Performance with respect to the syntactic unit of processing of the training datasets. Accuracy is the fraction of correctly ranked pairs of documents over the total number of pairs. (?Heads sentence? is the heads of NGs and VGs.)

Given the appropriate unit of granularity, we can consider the impact of semantic relations between surface realizations on coherence. For these experiments we use the synonym, hypernym, hyponym, and antonym relations in WordNet. The rationale for the consideration of semantic relations lies in the fact that the frequent use of the same words is usually deemed bad writing style. One therefore tends to observe the use of semantically similar terms in neighboring sentences. The results of using semantic relations for coherence rating are provided in Table 3. Synonym detection improves performance, while the other units provide poorer performance. This suggests that the hypernym and hyponym relations tend to over-generalize in the semantics.

The third category of features investigated is the temporal ordering of sentences; we use VerbOcean to obtain the temporal precedence between two events. One would expect events to be described ei-

Syntactic Unit	*Processing*	*Accuracy*	
		Acc	Earth
head NG	synonyms	82.37	59.40
	hypernyms	76.98	61.02
	hyponyms	81.59	59.14
	antonyms	74.20	48.07
	combines	70.84	56.51
head VG	synonyms	54.19	70.80
	hypernyms	53.36	60.54
	hyponyms	55.27	68.32
	antonyms	47.45	63.91
	combines	49.73	66.77

Table 3: The impact of WordNet on sentence ordering accuracy

Temporal Ordering	*Accuracy*	
	Acc	Earth
Precedence Ordering	60.41	47.09
Reverse Ordering	39.59	52.61
Precedence w/ matching NG	62.65	57.52
Reverse w/ matching NG	37.35	42.48

Table 4: The impact of the VerbOcean ?happens-before? temporal precedence relation on accuracy on the training datasets

ther in chronological order or in its reverse. While the former ordering represents a factual account of some sequence of events, the latter corresponds to newswire-style texts, which present the most important event(s) first, even though they may derive from previous events.

Table 4 provides the results of the experiments with temporal orderings. The first two rows validate the ordering of the events, while the latter two require the corresponding sentences to have a noun group in common in order to increase the likelihood that two events are related. The results clearly show that there is potential in the direct ordering of events. This suggests that sentence ordering can to some degree be achieved using simple temporal precedence orderings in a domain-independent way. This holds despite the results indicating that the features work better for sequences of events (as in the accident dataset) as opposed to accounts of the results of some event(s) (as in the earthquake dataset).

Range	*Accuracy*	
	Acc	Earth
2 occ. in 2 sent.	80.57	50.11
2 occ. in 3 sent.	73.17	45.43
3 occ. in 3 sent.	71.35	52.81
2 occ. in 4 sent.	66.95	50.41
3 occ. in 4 sent.	69.38	41.61
4 occ. in 4 sent.	71.93	58.97
2 occ. in 5 sent.	61.48	66.25
3 occ. in 5 sent.	68.59	42.33
4 occ. in 5 sent.	65.77	40.75
5 occ. in 5 sent.	81.39	62.40
sim. w/ sent. 1 sent. away	83.39	71.94
sim. w/ sent. 2 sent. away	60.44	67.52
sim. w/ sent. 3 sent. away	52.28	54.65
sim. w/ sent. 4 sent. away	49.65	44.50
sim. w/ sent. 5 sent. away	43.68	52.11

Table 5: Effect of longer range relations on coherence accuracy

The final category of features investigates the degree to which relations between sentences other than directly subsequent sentences are relevant. To this end, we explore two different approaches. The first set of features considers the distribution of entities within a fixed set of sentences, and captures in how many different sentences the entities occur. The resulting score is the number of times the entities occur in N out of M sentences. The second set only considers the similarity score from the current sentence and the other sentences within a certain range from the current sentence. The score of this feature is the sum of the individual similarities. Table 5 clearly confirms that longer range relations are relevant to the assessment of the coherence of text. An interesting difference between the two approaches is that sentence similarity only provides good results for neighboring sentences or sentences only one sentence apart, while the occurrence-counting method also works well over longer ranges.

Having evaluated the potential contributions of the individual features and their modeling, we now use SVMs to combine the features into one comprehensive measure. Given the indications from the foregoing experiments, the results in Table 6 are disappointing. In particular, the performance on the

Combination	*Accuracy*	
	Acc	Earth
Chunk+Temp+WN+LongRange+	83.11	54.88
Chunk+Temp+WN+LongRange-	77.67	62.76
Chunk+Temp+WN-LongRange+	74.17	59.28
Chunk+Temp+WN-LongRange-	68.15	63.55
Chunk+Temp-WN+LongRange+	*86.88*	*63.83*
Chunk+Temp-WN+LongRange-	80.19	59.43
Chunk+Temp-WN-LongRange+	76.63	60.86
Chunk+Temp-WN-LongRange-	64.43	60.94
NG Similarity w/ Synonyms	85.90	63.55
Coreference+Syntax+Salience+	90.4	87.2
Coreference-Syntax+Salience+	89.9	83.0
HMM-based Content Models	75.8	88.0
Latent Semantic Analysis	87.3	81.0

Table 6: Comparison of the developed model with other state-of-the-art systems. Coreference+Syntax+Salience+ and Coreference?Syntax+Salience+ are the Barzilay and Lapata (2008) model, HMM-based Content Models is the Barzilay and Lee (2004) paper and Latent Semantic Analysis is the Barzilay and Lapata (2008) implementation of Peter W. Foltz and Landauer (1998). The results of these systems are reproduced from Barzilay and Lapata (2008). (Temp = Temporal; WN = WordNet)

Coreference+Syntax+Salience+

Train \ Test	Earthquakes	Accidents
Earthquakes	87.3	67.0
Accidents	69.7	90.4

HMM-based Content Models

Train \ Test	Earthquakes	Accidents
Earthquakes	88.0	31.7
Accidents	60.3	75.8

Chunk+Temporal-WordNet+LongRange+

Train \ Test	Earthquakes	Accidents
Earthquakes	63.83	86.63
Accidents	64.19	86.88

Table 7: Cross-Training between Accident and Earthquake datasets. The results for Coreference+Syntax+Salience+ and HMM-Based Content Models are reproduced from Barzilay and Lapata (2008).

earthquake dataset is below standard. However, it seems that sentence ordering in that set is primarily defined by topics, as only content models perform well. (Barzilay and Lapata (2008) only perform well when using their coreference module, which determines antecedents based on the identified coreferences in the *original* sentence ordering, thereby biasing their orderings towards the correct ordering.) Longer range and WordNet relations together (Chunk+Temp-WN+LongRange+) achieve the best performance. The corresponding configuration is also the only one that achieves reasonable performance when compared with other systems.

4 Experiment 2

As stated, the ultimate goal of the models presented in this paper is the application of sentence ordering to automatically generated summaries. It is, in this regard, important to distinguish coherence as studied in Experiment 1 and coherence in the context of automatic summarization. Namely, for newswire summarization systems, the topics of the documents are unknown at the time of training. As a result, model performance on out-of-domain texts is important for summarization. Experiment 2 seeks to evaluate how well our model performs in such cases. To this end, we carry out two sets of tests. First, we cross-train the models between the accident and earthquake datasets to determine system performance in unseen domains. Second, we use the dataset based on the DUC 2005 model summaries to investigate whether our model's performance on unseen topics reaches a plateau after training on a particular number of different topics.

Surprisingly, the results are rather good, when compared to the poor results in part of the previous experiment (Table 7). In fact, model performance is nearly independent of the training topic. Nevertheless, the results on the earthquake test set indicate that our model is missing essential components for the correct prediction of sentence orderings on this set. When compared to the results obtained by Barzilay and Lapata (2008) and Barzilay and Lee (2004), it would appear that direct sentence-to-sentence similarity (as suggested by the Barzilay and Lapata baseline score) or capturing topic sequences are essential for acquiring the correct sequence of sentences in the earthquake dataset.

The final experimental setup applies the best

Different Topics	*Training Pairs*	*Accuracy*
2	160	55.17
4	420	63.54
6	680	65.20
8	840	65.57
10	1,100	64.80
15	1,500	64.93
20	2,100	64.87
25	2,700	64.94
30	3,300	65.61

Table 8: Accuracy on 20 test topics (2,700 pairs) with respect to the number of topics used for training using the model Chunk+Temporal-WordNet+LongRange+

model (Chunk+Temporal-WordNet+LongRange+) to the summarization dataset and evaluates how well the model generalises as the number of topics in the training dataset increases. The results - provided in Table 8 - indicate that very little training data (both regarding the number of pairs and the number of different topics) is needed. Unfortunately, they also suggest that the DUC summaries are more similar to the earthquake than to the accident dataset.

5 Conclusions

This paper investigated the effect of different features on sentence ordering. While a set of features has been identified that works well individually as well as in combination on the accident dataset, the results on the earthquake and DUC 2005 datasets are disappointing. Taking into account the performance of content models and the baseline of the Barzilay and Lapata (2008) model, the most convincing explanation is that the sentence ordering in the earthquake datasets is based on some sort of topic notion, providing a variety of possible antecedents between which our model is thus far unable to distinguish without resorting to the original (correct) ordering. Future work will have to concentrate on this aspect of sentence ordering, as it appears to coincide with the structure of the summaries for the DUC 2005 dataset.

References

Barzilay, R. (2003). *Information fusion for multi-document summarization: paraphrasing and generation*. Ph. D. thesis, Columbia University.

Barzilay, R. and M. Lapata (2008). Modeling local coherence: An entity-based approach. *Comput. Linguist. 34*, 1–34.

Barzilay, R. and L. Lee (2004). Catching the drift: probabilistic content models, with applications to generation and summarization. In *Proceedings of HLT-NAACL 2004*.

Bollegala, D., N. Okazaki, and M. Ishizuka (2006). A bottom-up approach to sentence ordering for multi-document summarization. In *Proceedings of ACL-44*.

Chklovski, T. and P. Pantel (2004). Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of EMNLP 2004*.

Dang, H. (2005). Overview of duc 2005.

Fellbaum, C. (Ed.) (1998). *WordNet An Electronic Lexical Database*. The MIT Press.

Grover, C. and R. Tobin (2006). Rule-based chunking and reusability. In *Proceedings of LREC 2006*.

Ji, P. D. and S. Pulman (2006). Sentence ordering with manifold-based classification in multi-document summarization. In *Proceedings of EMNLP 2006*.

Joachims, T. (2002). Evaluating retrieval performance using clickthrough data. In *Proceedings of the SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval*.

Karamanis, N. (2004). Evaluating centering for sentence ordering in two new domains. In *Proceedings of the NAACL 2004*.

Lapata, M. (2003). Probabilistic text structuring: Experiments with sentence ordering. In *Proc. of ACL 2003*.

Minnen, G., C. J. and D. Pearce (2000). Robust, applied morphological generation. In *Proceedings of the 1st International Natural Language Generation Conference*.

Peter W. Foltz, W. K. and T. K. Landauer (1998). Textual coherence using latent semantic analysis. *Discourse Processes 25*, 285–307.

Towards Unsupervised Recognition of Dialogue Acts

Nicole Novielli
Dept. of Informatics, University of Bari
via Orabona 4
I-70125 Bari, Italy
novielli@di.uniba.it

Carlo Strapparava
FBK-irst
via Sommarive, Povo
I-38050 Trento, Italy
strappa@fbk.eu

Abstract

When engaged in dialogues, people perform communicative actions to pursue specific communicative goals. Speech acts recognition attracted computational linguistics since long time and could impact considerably a huge variety of application domains. We study the task of automatic labeling dialogues with the proper dialogue acts, relying on empirical methods and simply exploiting lexical semantics of the utterances. In particular, we present some experiments in supervised and unsupervised framework on both an English and an Italian corpus of dialogue transcriptions. The evaluation displays encouraging results in both languages, especially in the unsupervised version of the methodology.

1 Introduction

People proceed in their conversations through a sequence of dialogue acts to yield some specific communicative goal. They can ask for information, agree or disagree with their partner, state some facts and express opinions.

Dialogue Acts (DA) attracted linguistics (Austin, 1962; Searle, 1969) and computational linguistics research (Core and Allen, 1997; Traum, 2000) since long time. With the advent of the Web, a large amount of material about natural language interactions (e.g. blogs, chats, conversation transcripts) has become available, raising the attractiveness of empirical methods analyses on this field. There is a large number of application domains that could benefit from automatically labeling DAs: e.g. conversational agents for monitoring and supporting human-human remote conversations, blogs, forums and chat logs analysis for opinion mining, interpersonal stances modeling by mean of conversational analysis, automatic meeting summarizations and so on. These applications require a deep understanding of the conversational structure and the ability of the system to understand who is telling what to whom.

This study defines a method for automatically labeling dialogues with the proper speech acts by relying on empirical methods. Even if prosody and intonation surely play a role (e.g. (Stolcke et al., 2000; Warnke et al., 1997)), nonetheless language and words are what the speaker uses to convey the communicative message and are just what we have at disposal when we consider texts found on the Web. Hence, we decided to simply exploit lexical semantics of the sentences. We performed some experiments in a supervised and unsupervised framework on both an English and an Italian corpora of dialogue transcriptions, achieving good results in all settings. Unsupervised performance is particularly encouraging, independently from the used language.

The paper is organized as follows. Section 2 gives a brief sketch of the NLP background on Dialogue Acts recognition. In Section 3 we introduce the English and Italian corpora of dialogues, their characteristics and DA labeling. In Section 4 we describe the preprocessing of the data sets. Then Section 5 explains the supervised and unsupervised settings, showing the experimental results obtained on the two corpora and providing an error analysis. Finally, in Section 6 we conclude the paper with a brief discussion and some directions for future work.

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 84–89,
Boulder, Colorado, June 2009.

Speaker	Dialogue Act	Utterance
A	OPENING	*Hello Ann.*
B	OPENING	Hello Chuck.
A	STATEMENT	*Uh, the other day, I attended a conference here at Utah State University on recycling*
A	STATEMENT	*and, uh, I was kind of interested to hear cause they had some people from the EPA and lots of different places, and, uh, there is going to be a real problem on solid waste.*
B	OPINION	Uh, I didn't think that was a new revelation.
A	AGREE /ACCEPT	*Well, it's not too new.*
B	INFO-REQUEST	So what is the EPA recommending now?

Table 1: An excerpt from the Switchboard corpus

2 Background

A DA can be identified with the communicative goal of a given utterance (Austin, 1962). Researchers use different labels and definitions to address this concept: *speech act* (Searle, 1969), *adjacency pair part* (Schegloff, 1968) (Sacks et al., 1974), *game move* (Power, 1979)

Traditionally, the NLP community has employed DA definitions with the drawback of being domain or application oriented. Recently some efforts have been made towards unifying the DA annotation (Traum, 2000). In the present study we refer to a domain-independent framework for DA annotation, the DAMSL architecture (Dialogue Act Markup in Several Layers) by (Core and Allen, 1997).

Recently, the problem of DA recognition has been addressed with promising results: Poesio and Mikheev (1998) combine expectations about the next likely dialogue 'move' with information derived from the speech signal features; Stolcke et al. (2000) employ a discourse grammar, formalized in terms of Hidden Markov Models, combining also evidences about lexicon and prosody; Keizer et al. (2002) make use of Bayesian networks for DA recognition in dutch dialogues; Grau et al. (2004) consider naive Bayes classifiers as a suitable approach to the DA classification problem; a partially supervised framework has also been explored by Venkataraman et al. (2005)

Regardless of the model they use (discourse grammars, models based on word sequences or on the acoustic features or a combination of all these) the mentioned studies are developed in a supervised framework. In this paper, one goal is to explore also the use of a fully unsupervised methodology.

3 Data Sets

In the experiments of the present paper we exploit two corpora, both annotated with DAs labels. We aim at developing a recognition methodology as general as possible, so we selected corpora which are different in content and language: the Switchboard corpus (Godfrey et al., 1992), a collection of transcriptions of spoken English telephone conversations about general interest topics, and an Italian corpus of dialogues in the healthy-eating domain (Clarizio et al., 2006).

In this section we describe the two corpora, their features, the set of labels used for annotating the dialogue acts with their distributions and the data preprocessing.

3.1 Description

The Switchboard corpus is a collection of English human-human telephone conversations (Godfrey et al., 1992) between couples of randomly selected strangers. They were asked to choose one general interest topic and to talk informally about it. Full transcripts of these dialogues are distributed by the Linguistic Data Consortium. A part of this corpus is annotated (Jurafsky et al., 1997) with DA labels (overall 1155 conversations, for a total of 205,000 utterances and 1.4 million words)[1]. Table 1 shows a short sample fragments of dialogues from the Switchboard corpus.

The Italian corpus had been collected in the scope of some previous research about Human-ECA interaction. A Wizard of Oz tool was employed (Clarizio et al., 2006) and during the interaction, a conversational agent (i.e. the 'wizard') played the role of

[1] `ftp.ldc.upenn.edu/pub/ldc/public\_data/swb1\_dialogact\_annot.tar.gz`

Label	Description	Example	Italian	English
INFO-REQUEST	Utterances that are pragmatically, semantically, and syntactically questions	*'What did you do when your kids were growing up?'*	34%	7%
STATEMENT	Descriptive, narrative, personal statements	*'I usually eat a lot of fruit'*	37%	57%
S-OPINION	Directed opinion statements	*'I think he deserves it.'*	6%	20%
AGREE-ACCEPT	Acceptance of a proposal, plan or opinion	*'That's right'*	5%	9%
REJECT	Disagreement with a proposal, plan, or opinion	*'I'm sorry no'*	7%	.3%
OPENING	Dialogue opening or self-introduction	*'Hello, my name is Imma'*	2%	.2%
CLOSING	Dialogue closing (e.g. farewell and wishes)	*'It's been nice talking to you.'*	2%	2%
KIND-ATT	Kind attitude (e.g. thanking and apology)	*'Thank you very much.'*	9%	.1%
GEN-ANS	Generic answers to an Info-Request	*'Yes', 'No', 'I don't know'*	4%	4%
total cases			1448	131,265

Table 2: The set of labels employed for Dialogue Acts annotation and their distribution in the two corpora

an artificial therapist. The users were free to interact with it in natural language, without any particular constraint. This corpus is about healthy eating and contains (overall 60 dialogues, 1448 users' utterances and 15,500 words).

3.2 Labelling

Both corpora are annotated following the Dialogue Act Markup in Several Layers (DAMSL) annotation scheme (Core and Allen, 1997). In particular the Switchboard corpus employs a revision (Jurafsky et al., 1997).[2]

Table 2 shows the set of labels employed with their definitions, examples and distributions in the two data sets. The categories maintain the DAMSL main characteristic of being domain-independent and can be easily mapped back into SWBD-DAMSL ones, and maintain their original semantics. Thus, the original SWBD-DAMSL annotation had been automatically converted into the categories included in our markup language.[3]

4 Data preprocessing

To reduce the data sparseness, we used a POS-tagger and morphological analyzer (Pianta et al., 2008) for preprocessing both corpora. So we considered lemmata instead of tokens in the format *lemma#POS*. In addition, we augment the features of each sentence with a set of linguistic markers, defined according to the semantic of the DA categories. We hypothesize, in fact, these features could play an important role in defining the linguistic profile of each DA. The addition of these markers is performed automatically, by just exploiting the output of the POS-tagger and of the morphological analyzer, according to the following rules:

- **WH-QTN**, used whenever an interrogative determiner (e.g. 'what') is found, according to the output of the POS-tagger;
- **ASK-IF**, used whenever an utterance presents the pattern of a 'Yes/No' question. ASK-IF and WH-QTN markers are supposed to be relevant for the INFO-REQUEST category;
- **I-PERS**, used for all declarative utterances whenever a verb is in the first person form, singular or plural (relevant for the STATEMENT);
- **COND**, used for conditional form is detected.
- **SUPER**, used for superlative adjectives.
- **AGR-EX**, used whenever an agreement expression (e.g.'You're right', 'I agree') is detected (relevant for AGREE-ACCEPT);
- **NAME**, used whenever a proper name follows a self-introduction expression (e.g. 'My name is') (relevant for the OPENING);
- **OR-CLAUSE**, used for or-clauses, that is utterance starting by 'or' (should be helpful for the characterization of the INFO-REQUEST);
- **VB**, used only for the Italian, when a dialectal form of agreement expression is detected.

5 Dialogue Acts Recognition

We conducted some experiments both in a supervised and unsupervised settings.

[2]The SWBD-DAMSL modifies the original DAMSL framework by further specifying some categories or by adding extra features (mainly prosodic) which were not originally included in the scheme.

[3]Also we did not consider the utterances formed only by non-verbal material (e.g. laughter).

5.1 Supervised

Regarding the supervised experiments, we used Support Vector Machines (Vapnik, 1995), in particular SVM-light package (Joachims, 1998) under its default configuration. We randomly split the two corpora into 80/20 training/test partitions. SVMs have been used in a large range of problems, including text classification, image recognition tasks, bioinformatics and medical applications, and they are regarded as the state-of-the-art in supervised learning. We got .71 and .77 of F1 measures respectively for the Italian and English corpus. Table 4 reports the performance for each direct act.

5.2 Unsupervised

It is not always easy to collect large training, partly because of manual labeling effort and moreover because often it is not possible to find it.

Schematically, our unsupervised methodology is: (i) building a semantic similarity space in which words, set of words, text fragments can be represented homogeneously, (ii) finding seeds that properly represent dialogue acts and considering their representations in the similarity space, and (iii) checking the similarity of the utterances.

To get a similarity space with the required characteristics, we used Latent Semantic Analysis (LSA), a corpus-based measure of semantic similarity proposed by Landauer (Landauer et al., 1998). In LSA, term co-occurrences in a corpus are captured by means of a dimensionality reduction operated by a singular value decomposition (SVD) on the term-by-document matrix $\mathbf{T}$ representing the corpus.

SVD decomposes the term-by-document matrix $\mathbf{T}$ into three matrices $\mathbf{T} = \mathbf{U}\mathbf{\Sigma_k}\mathbf{V}^T$ where $\mathbf{\Sigma_k}$ is the diagonal $k \times k$ matrix containing the k singular values of $\mathbf{T}$, $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_k$, and $\mathbf{U}$ and $\mathbf{V}$ are column-orthogonal matrices. When the three matrices are multiplied together the original term-by-document matrix is re-composed. Typically we can choose $k' \ll k$ obtaining the approximation $\mathbf{T} \simeq \mathbf{U}\mathbf{\Sigma_{k'}}\mathbf{V}^T$.

LSA can be viewed as a way to overcome some of the drawbacks of the standard vector space model (sparseness and high dimensionality). In fact, the LSA similarity is computed in a lower dimensional space, in which second-order relations among terms and texts are exploited. The similarity in the resulting vector space is then measured with the standard cosine similarity. Note also that LSA yields a vector space model that allows for a *homogeneous* representation (and hence comparison) of words, sentences, and texts. For representing a word set or a sentence in the LSA space we use the *pseudo-document* representation technique, as described by Berry (1992). In practice, each text segment is represented in the LSA space by summing up the normalized LSA vectors of all the constituent words, using also a *tf.idf* weighting scheme (Gliozzo and Strapparava, 2005).

Label	Seeds
INFO-REQ	WH-QTN, Question_Mark, ASK-IF, huh
STATEMENT	I-PERS, I
S-OPINION	Verbs which directly express opinion or evaluation (guess, think, suppose, affect)
AGREE-ACC	AGR-EX, yep, yeah, absolutely, correct
REJECT	Verbs which directly express disagreement (disagree, refute)
OPENING	Greetings (hi, hello), words and markers related to self-introduction (name, NAME)
CLOSING	Interjections/exclamations ending discourse (alright, okeydoke), Expressions of thanking (thank) and farewell (bye, bye-bye, goodnight, goodbye)
KIND-ATT	Wishes (wish), apologies (apologize), thanking (thank) and sorry-for (sorry, excuse)
GEN-ANS	no, yes, uh-huh, nope

Table 3: The seeds for the unsupervised experiment

The methodology is completely unsupervised. We run the LSA using 400 dimensions (i.e. k', as suggested by (Landauer et al., 1998)) respectively on the English and Italian corpus, without any DA label information. Starting from a set of seeds (words) representing the communicative acts (see the complete sets in Table 3), we build the corresponding vectors in the LSA space and then we compare the utterances to find the communicative act with higher similarity. To compare with SVM, the performance is measured on the same test set partition used in the supervised experiment (Table 4).

We defined seeds by only considering the communicative goal and the specific semantic of every single DA, just avoiding as much as possible the overlapping between seeds groups. We wanted to design

	Italian						English					
	SVM			LSA			SVM			LSA		
Label	prec	rec	f1	prec	rec	f1	prec	rec	f1	prec	rec	f1
INFO-REQ	.92	.99	.95	.96	.88	.92	.92	.84	.88	.93	.70	.80
STATEMENT	.85	.68	.69	.76	.66	.71	.79	.92	.85	.70	.95	.81
S-OPINION	.28	.42	.33	.24	.42	.30	.66	.44	.53	.41	.07	.12
AGREE-ACC	.50	.80	.62	.56	.50	.53	.69	.74	.71	.68	.63	.65
REJECT	-	-	-	.09	.25	.13	-	-	-	.01	.01	.01
OPENING	.60	1.00	.75	.55	1.00	.71	.96	.55	.70	.20	.43	.27
CLOSING	.67	.40	.50	.25	.40	.31	.83	.59	.69	.76	.34	.47
KIND-ATT	.82	.53	.64	.43	.18	.25	.85	.34	.49	.09	.47	.15
GEN-ANS	.20	.63	.30	.27	.38	.32	.56	.25	.35	.54	.33	.41
micro	.71	.71	.71	.66	.66	.66	.77	.77	.77	.69	.69	.69

Table 4: Evaluation of the two methods on both corpora

an approach which is as general as possible, so we did not consider domain words. The seeds are the same for both languages, which is coherent with our goal of defining a language-independent method.

5.3 Experimental Results and Discussion

We evaluate the performance of our method in terms of precision, recall and f1-measure (see Table 4) according to the DA labels given by annotators in the datasets. As baselines we consider (i) most-frequent label assignment (respectively 37% for Italian, 57% for English) for the supervised setting, and (ii) random DA selection (11%) for the unsupervised one.

Results are quite satisfying (Table 4). In particular, the unsupervised technique is largely above the baselines, for both the Italian and the English experiments. The methodology is independent from the language and the domain: the Italian corpus is a collection of dialogue about a very restricted domain while the Switchboard conversations are essentially task-free. Moreover, in the unsupervised setting we use in practice the same seed definitions. Secondly, it is independent on the differences in the linguistic style due to the specific interaction scenario and input modality. Finally, the performance is not affected by the difference in size of the two data sets.

Error analysis. After conducting an error analysis, we noted that many utterances are misclassified as STATEMENT. One possible reason is that statements usually are quite long and there is a high chance that some linguistic markers that characterize other dialogue acts are present in those sentences. On the other hand, looking at the corpora we observed that many utterances which appear to be linguistically consistent with the typical structure of statements have been annotated differently, according to the actual communicative role they play. For similar reasons, we observed some misclassification of S-OPINION as STATEMENT. The only significative difference between the two labels seems to be the wider usage of 'slanted' and affectively loaded lexicon when conveying an opinion. Another cause of confounding is the confusion among the backchannel labels (GEN-ANS, AGREE-ACC and REJECT) due to the inherent ambiguity of common words like *yes*, *no*, *yeah*, *ok*.

Recognition of such cases could be improved (i) by enabling the classifiers to consider not only the lexical semantics of the given utterance (local context) but also the knowledge about a wider context window (e.g. the previous *n* utterances), (ii) by enriching the data preprocessing (e.g. by exploiting information about lexicon polarity and subjectivity parameters). We intend to follow both these directions in our future research.

6 Conclusions and Future Work

This study aims at defining a method for Dialogue Acts recognition by simply exploiting the lexical semantics of dialogue turns. The technique had to be independent from some important features of the corpus being used such as domain, language, size, interaction scenario. In a long-term perspective, we will employ the technique in conversational analysis for user attitude classification (Martalo et al., 2008).

The methodology starts with automatically en-

riching the corpus with additional features, such as linguistic markers. Then the unsupervised case consists of defining a very simple and intuitive set of seeds that profiles the specific dialogue acts, and subsequently performing a similarity analysis in a latent semantic space. The performance of the unsupervised experiment has been compared with a supervised state-of-art technique such as Support Vector Machines, and the results are quite encouraging.

Regarding future developments, we will investigate how to include in the framework a wider context (e.g. the previous *n* utterances), and the introduction of new linguistic markers by enriching the preprocessing techniques. In particular, it would be interesting to exploit the role of slanted or affective-loaded lexicon to deal with the misclassification of opinions as statements. Along this perspective, DA recognition could serve also as a basis for conversational analysis aimed at improving a fine-grained opinion mining in dialogues.

References

J. Austin. 1962. *How to do Things with Words*. Oxford University Press, New York.

M. Berry. 1992. Large-scale sparse singular value computations. *International Journal of Supercomputer Applications*, 6(1).

G. Clarizio, I. Mazzotta, N. Novielli, and F. deRosis. 2006. Social attitude towards a conversational character. In *Proceedings of the 15th IEEE International Symposium on Robot and Human Interactive Communication*, pages 2–7, Hatfield, UK, September.

M. Core and J. Allen. 1997. Coding dialogs with the DAMSL annotation scheme. In *Working Notes of the AAAI Fall Symposium on Communicative Action in Humans and Machines*, Cambridge, MA.

A. Gliozzo and C. Strapparava. 2005. Domains kernels for text categorization. In *Proceedengs of (CoNLL-2005)*, University of Michigan, Ann Arbor, June.

J. Godfrey, E. Holliman, and J. McDaniel. 1992. SWITCHBOARD: Telephone speech corpus for research and development. In *Proceedings of ICASSP-92*, pages 517–520, San Francisco, CA. IEEE.

S. Grau, E. Sanchis, M. J. Castro, and D. Vilar. 2004. Dialogue act classification using a bayesian approach. In *Proceedings of SPECOM-04*, pages 495–499, Saint-Petersburg, Russia, September.

T. Joachims. 1998. Text categorization with Support Vector Machines: learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*.

D. Jurafsky, E. Shriberg, and D. Biasca. 1997. Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual, draft 13. Technical Report 97-01, University of Colorado.

S. Keizer, R. op den Akker, and A. Nijholt. 2002. Dialogue act recognition with bayesian networks for dutch dialogues. In K. Jokinen and S. McRoy, editors, *Proceedings 3rd SIGdial Workshop on Discourse and Dialogue*, pages 88–94, Philadelphia, PA, July.

T. K. Landauer, P. Foltz, and D. Laham. 1998. Introduction to latent semantic analysis. *Discourse Processes*, 25.

A. Martalo, N. Novielli, and F. deRosis. 2008. Attitude display in dialogue patterns. In *AISB 2008 Convention on Communication, Interaction and Social Intelligence*, Aberdeen, Scotland, April.

E. Pianta, C. Girardi, and R. Zanoli. 2008. The TextPro tool suite. In *Proceedings of LREC*, Marrakech (Morocco), May.

M. Poesio and A. Mikheev. 1998. The predictive power of game structure in dialogue act recognition: Experimental results using maximum entropy estimation. In *Proceedings of ICSLP-98*, Sydney, December.

R. Power. 1979. The organisation of purposeful dialogues. *Linguistics*, 17:107–152.

H. Sacks, E. Schegloff, and G. Jefferson. 1974. A simplest systematics for the organization of turn-taking for conversation. *Language*, 50(4):696–735.

E. Schegloff. 1968. Sequencing in conversational openings. *American Anthropologist*, 70:1075–1095.

J. Searle. 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, Cambridge, London.

A. Stolcke, N. Coccaro, R. Bates, P. Taylor, C. Van Ess-Dykema, K. Ries, E. Shriberg, D. Jurafsky, R. Martin, and M. Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–373.

D. Traum. 2000. 20 questions for dialogue act taxonomies. *Journal of Semantics*, 17(1):7–30.

V. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer-Verlag.

A. Venkataraman, Y. Liu, E. Shriberg, and A. Stolcke. 2005. Does active learning help automatic dialog act tagging in meeting data? In *Proceedings of EUROSPEECH-05*, Lisbon, Portugal.

V. Warnke, R. Kompe, H. Niemann, and E. Nöth. 1997. Integrated dialog act segmentation and classification using prosodic features and language models. In *Proceedings of 5th European Conference on Speech Communication and Technology*, volume 1, pages 207–210, Rhodes, Greece.

Modeling Letter-to-Phoneme Conversion as a Phrase Based Statistical Machine Translation Problem with Minimum Error Rate Training

Taraka Rama, Anil Kumar Singh, Sudheer Kolachina
Language Technologies Research Centre,
IIIT, Hyderabad, India.
{taraka@students,anil@research,sudheer.kpg08@research}.iiit.ac.in

Abstract

Letter-to-phoneme conversion plays an important role in several applications. It can be a difficult task because the mapping from letters to phonemes can be many-to-many. We present a language independent letter-to-phoneme conversion approach which is based on the popular phrase based Statistical Machine Translation techniques. The results of our experiments clearly demonstrate that such techniques can be used effectively for letter-to-phoneme conversion. Our results show an overall improvement of 5.8% over the baseline and are comparable to the state of the art. We also propose a measure to estimate the difficulty level of L2P task for a language.

1 Introduction

Letter-to-phoneme (L2P) conversion can be defined as the task of predicting the pronunciation of a word given its orthographic form (Bartlett et al., 2008).The pronunciation is usually represented as a sequence of phonemes. Letter-to-phoneme conversion systems play a very important role in spell checkers (Toutanova and Moore, 2002), speech synthesis systems (Schroeter et al., 2002) and transliteration (Sherif and Kondrak, 2007). Letter-to-phoneme conversion systems may also be effectively used for cognate identification and transliteration. The existing cognate identification systems use the orthographic form of a word as the input. But we know that the correspondence between written and spoken forms of words can be quite irregular as is the case in English. Even in other languages with supposedly regular spellings, this irregularity exists owing to linguistic phenomena like borrowing and language variation. Letter-to-phoneme conversion systems can facilitate the task of cognate identification by providing a language independent transcription for any word.

Until a few years ago, letter-to-phoneme conversion was performed considering only one-one correspondences (Black et al., 1998; Damper et al., 2004). Recent work uses many-to-many correspondences (Jiampojamarn et al., 2007) and reports significantly higher accuracy for Dutch, German and French. The current state of the art systems give as much as 90% (Jiampojamarn et al., 2008) accuracy for languages like Dutch, German and French. However, accuracy of this level is yet to be achieved for English.

Rule-based approaches to the problem of letter-to-phoneme conversion although appealing, are impractical as the number of rules for a particular language can be very high (Kominek and Black, 2006). Alternative approaches to this problem are based on machine learning and make use of resources such as pronunciation dictionaries. In this paper, we present one such machine learning based approach wherein we envisage this problem as a Statistical Machine Translation (SMT) problem.

The outline of the paper is as follows. Section 2 presents a brief summary of the related work done in L2P conversion. Section 3 describes our model and the techniques devised for optimizing the performance. Section 4 describes the letter-to-phoneme alignment. The description of the results and experiments and a new technique for estimating the diffi-

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 90–95,
Boulder, Colorado, June 2009.

culty level of L2P task have been given in Section 5. Error analysis is presented in Section 6. Finally we conclude with a summary and suggest directions for future work.

2 Related Work

In the letter-to-phoneme conversion task, a single letter can map to multiple phonemes [x → ks] and multiple letters can generate a single phoneme. A letter can also map to a null phoneme [e → φ] and vice-versa. These examples give a glimpse of why the task is so complex and a single machine learning technique may not be enough to solve the problem. A overview of the literature supports this claim.

In older approaches, the alignment between the letters and phonemes was taken to be one-to-one (Black et al., 1998) and the phoneme was predicted for every single letter. But recent work (Bisani and Ney, 2002; Jiampojamarn et al., 2007) shows that multiple letter-to-phoneme alignments perform better than single letter to phoneme alignments. The problem can be either viewed as a multi-class classifier problem or a structure prediction problem. In structure prediction, the algorithm takes the previous decisions as the features which influence the current decision.

In the classifier approach, only the letter and its context are taken as features. Then, either multiclass decision trees (Daelemans and van den Bosch, 1997) or instance based learning as in (van den Bosch and Daelemans, 1998) is used to predict the class, which in this case is a phoneme. Some of these methods (Black et al., 1998) are not completely automatic and need an initial handcrafted seeding to begin the classification.

Structure prediction is like a tagging problem where HMMs (Taylor, 2005) are used to model the problem. Taylor claims that except for a preprocessing step, it is completely automatic. The whole process is performed in a single step. The results are poor, as reasoned in (Jiampojamarn et al., 2008) due to the emission probabilities not being informed by the previous letter's emission probabilities. Pronunciation by Analogy (PbA) is a data-driven method (Marchand and Damper, 2000) for letter-to-phoneme conversion which is used again by Damper et al (2004). They simply use an Expectation-Maximisation (EM) like algorithm for aligning the letter-phoneme pairs in a speech dictionary. They claim that by integrating the alignments induced by the algorithm into the PbA system, they were able to improve the accuracy of the pronunciation significantly. We also use the many-to-many alignment approach but in a different way and obtained from a different source.

The recent work of Jiampojamarn et al (2007) combines both of the above approaches in a very interesting manner. It uses an EM like algorithm for aligning the letters and phonemes. The algorithm allows many-to-many alignments between letters and phonemes. Then there is a letter chunking module which uses instance-based training to train on the alignments which have been obtained in the previous step. This module is used to guess the possible letter chunks in every word. Then a local phoneme predictor is used to guess the phonemes for every letter in a word. The size of the letter chunk could be either one or two. Only one candidate for every word is allowed. The best phoneme sequence is obtained by using Viterbi search.

An online model MIRA (Crammer and Singer, 2003) which updates parameters is used for the L2P task by Jiampojamarn et al (2008). The authors unify the steps of letter segmentation, phoneme prediction and sequence modeling into a single module. The phoneme prediction and sequence modeling are considered as tagging problems and a Perceptron HMM (Collins, 2002) is used to model it. The letter segmenter module is replaced by a monotone phrasal decoder (Zens and Ney, 2004) to search for the possible substrings in a word and output the n-best list for updating MIRA. Bisani and Ney (2002) take the joint multigrams of graphemes and phonemes as features for alignment and language modeling for phonetic transcription probabilities. A hybrid approach similar to this is by (van den Bosch and Canisius, 2006).

In the next section we model the problem as a Statistical Machine Translation (SMT) task.

3 Modeling the Problem

Assume that given a word, represented as a sequence of letters $\mathbf{l} = l_1^J = l_1 ... l_j ... l_J$, needs to be transcribed as a sequence of phonemes, represented as $\mathbf{f}$

$= f_1^I = f_1 ... f_i ... f_I$. The problem of finding the best phoneme sequence among the candidate translations can be represented as:

$$\mathbf{f}_{best} = \arg\max_{\mathbf{f}} \{\Pr(\mathbf{f} \mid \mathbf{l})\} \quad (1)$$

We model the problem of letter to phoneme conversion based on the noisy channel model. Reformulating the above equation using Bayes Rule:

$$\mathbf{f}_{best} = \arg\max_{\mathbf{f}} p(\mathbf{l} \mid \mathbf{f})\, p(\mathbf{f}) \quad (2)$$

This formulation allows for a phoneme n-gram model $p(\mathbf{f})$ and a transcription model $p(\mathbf{l} \mid \mathbf{f})$. Given a sequence of letters **l**, the argmax function is a search function to output the best phonemic sequence. During the decoding phase, the letter sequence **l** is segmented into a sequence of *K* letter segments $\bar{l}_1^K$. Each segment $\bar{l}_k$ in $\bar{l}_1^K$ is transcribed into a phoneme segment $\bar{f}_k$. Thus the best phoneme sequence is generated from left to right in the form of partial translations. By using an *n*-gram model p_{LM} as the language model, we have the equations:

$$\mathbf{f}_{best} = \arg\max_{\mathbf{f}} p(\mathbf{l} \mid \mathbf{f})\, p_{LM} \quad (3)$$

with $p(\mathbf{l} \mid \mathbf{f})$ written as

$$p(\bar{l}_1^K \mid \bar{f}_1^K) = \prod_{k=1}^{K} \Phi(\bar{l}_k \mid \bar{f}_k) \quad (4)$$

From the above equation, the best phoneme sequence is obtained based on the product of the probabilities of transcription model and the probabilities of a language model and their respective weights. The method for obtaining the transcription probabilities is described briefly in the next section. Determining the best weights is necessary for obtaining the right phoneme sequence. The estimation of the models' weights can be done in the following manner.

The posterior probability $\Pr(\mathbf{f} \mid \mathbf{l})$ can also be directly modeled using a log-linear model. In this model, we have a set of M feature functions $h_m(\mathbf{f}, \mathbf{l}), m = 1...M$. For each feature function there exists a weight or model parameter $\lambda_m, m = 1...M$. Thus the posterior probability becomes:

$$\Pr(\mathbf{f} \mid \mathbf{l}) = p_{\lambda_1^M}(\mathbf{f} \mid \mathbf{l}) \quad (5)$$

$$= \frac{\exp\left[\Sigma_{m=1}^{M} \lambda_m h_m(\mathbf{f}, \mathbf{l})\right]}{\sum_{f_1^I} \exp\left[\Sigma_{m=1}^{M} \lambda_m h_m(f_1^I, \mathbf{l})\right]} \quad (6)$$

with the denominator, a normalization factor that can be ignored in the maximization process.

The above modeling entails finding the suitable model parameters or weights which reflect the properties of our task. We adopt the criterion followed in (Och, 2003) for optimising the parameters of the model. The details of the solution and proof for the convergence are given in Och (2003). The models' weights, used for the L2P task, are obtained from this training.

4 Letter-to-Phoneme Alignment

We used GIZA++ (Och and Ney, 2003), an open source toolkit, for aligning the letters with the phonemes in the training data sets. In the context of SMT, say English-Spanish, the parallel corpus is aligned bidirectionally to obtain the two alignments. The IBM models give only one-to-one alignments between words in a sentence pair. So, GIZA++ uses some heuristics to refine the alignments (Och and Ney, 2003).

In our input data, the source side consists of grapheme (or letter) sequences and the target side consists of phoneme sequences. Every letter or grapheme is treated as a single 'word' for the GIZA++ input. The transcription probabilities can then be easily learnt from the alignments induced by GIZA++, using a scoring function (Koehn et al., 2003). Figure 1 shows the alignments induced by GIZA++ for the example words which are mentioned by Jiampojamarn et al (2007). In this figure, we only show the alignments from graphemes to phonemes.

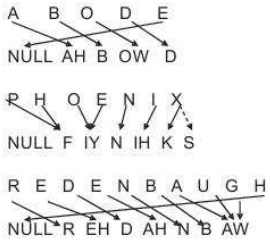

Figure 1: Example Alignments from GIZA++

5 Evaluation

We evaluated our models on the English CMUDict, French Brulex, German Celex and Dutch Celex speech dictionaries. These dictionaries are available for download on the website of PROANALSYL[1] Letter-to-Phoneme Conversion Challenge. Table 1 shows the number of words for each language. The datasets available at the website were divided into 10 folds. In the process of preparing the datasets we took one set for test, another for developing our parameters and the remaining 8 sets for training. We report our results in word accuracy rate, based on 10-fold cross validation, with mean and standard deviation.

Language	Datasets	Number of Words
English	CMUDict	112241
French	Brulex	27473
German	Celex	49421
Dutch	Celex	116252

Table 1: Number of words in each Dataset

We removed the one-to-one alignments from the corpora and induced our own alignments using GIZA++. We used minimum error rate training (Och, 2003) and the A* beam search decoder implemented by Koehn (Koehn et al., 2003). All the above tools are available as parts of the MOSES (Koehn et al., 2007) toolkit.

5.1 Exploring the Parameters

The parameters which have a major influence on the performance of a phrase-based SMT model are the alignment heuristics, the maximum phrase length (MPR) and the order of the language model (Koehn et al., 2003). In the context of letter to phoneme conversion, *phrase* means a sequence of letters or phonemes mapped to each other with some probability (i.e., the *hypothesis*) and stored in a phrase table. The *maximum phrase length* corresponds to the maximum number of letters or phonemes that a hypothesis can contain. Higher phrase length corresponds a larger phrase table during decoding.

We have conducted experiments to see which combination gives the best output. We initially trained the model with various parameters on the training data and tested for various values of the above parameters. We varied the maximum phrase length from 2 to 7. The language model was trained using SRILM toolkit (Stolcke, 2002). We varied the order of language model from 2 to 8. We also traversed the alignment heuristics spectrum, from the parsimonious *intersect* at one end of the spectrum through *grow, grow-diag, grow-diag-final, grow-diag-final-and* and *srctotgt* to the most lenient *union* at the other end. Our intuitive guess was that the best alignment heuristic would be *union*.

We observed that the best results were obtained when the language model was trained on 6-gram and the alignment heuristic was *union*. No significant improvement was observed in the results when the value of MPR was greater than 5. We have taken care such that the alignments are always monotonic. Note that the average length of the phoneme sequence was also 6. We adopted the above parameter settings for performing training on the input data.

5.2 System Comparison

We adopt the results given in (2007) as our baseline. We also compare our results with some other recent techniques mentioned in the Related Work section. Table 2 shows the results. As this table shows, our approach yields the best results in the case of German and Dutch. The word accuracy obtained for the German Celex and Dutch Celex dataset using our approach is higher than that of all the previous approaches listed in the table. In the case of English and French, although the baseline is achieved through our approach, the word accuracy falls short of being the best. However, it must also be noted that the dataset that we used for English is slightly larger than those of the other systems shown in the table.

We also observe that for an average phoneme accuracy of 91.4%, the average word accuracy is 63.81%, which corroborates the claim by Black et al (Black et al., 1998) that a 90% phoneme accuracy corresponds to 60% word accuracy.

5.3 Difficulty Level and Accuracy

We also propose a new language-independent measure that we call 'Weighted Symmetric Cross Entropy' (WSCE) to estimate the difficulty level of the L2P task for a particular language. The *weighted*

[1] http://www.pascal-network.org/Challenges/PRONALSYL/

Language	Dataset	Baseline	CART	1-1 Align	1-1 + CSIF	1-1 + HMM	M-M Align	M-M + HMM	MeR + A*
English	CMUDict	58.3±0.49	57.8	60.3±0.53	62.9±0.45	62.1±0.53	65.1±0.60	65.6±0.72	63.81±0.47
German	Celex	86.0±0.40	89.38	86.6±0.54	87.6±0.47	87.6±0.59	89.3±0.53	89.8±0.59	90.20±0.25
French	Brulex	86.3±0.67	-	87.0±0.38	86.5±0.68	88.2±0.39	90.6±0.57	90.9±0.45	86.71±0.52
Dutch	Celex	84.3± 0.34	-	86.6±0.36	87.5±0.32	87.6±0.34	91.1±0.27	91.4±0.24	91.63±0.24

Table 2: System Comparison in terms of word accuracies. **Baseline:**Results from PRONALSYS website. **CART:** CART Decision Tree System (Black et al., 1998). **1-1 Align, M-M align, HMM:** one-one alignments, many-many alignments, HMM with local prediction (Jiampojamarn et al., 2007). **CSIF:**Constraint Satisfaction Inference(CSIF) of(van den Bosch and Canisius, 2006). **MeR+A*:**Our approach with minimum error rate training and A* search decoder. "-" refers to no reported results.

SCE is defined as follows:

$$d_{sce_{wt}} = \sum r_t \left(p_l \, log \left(q_f\right) + q_f \, log \left(p_l\right)\right) \qquad (7)$$

where p and q are the probabilities of occurrence of letter (l) and phoneme (f) sequences, respectively. Also, r_t corresponds to the conditional probability $p(f \mid l)$. This transcription probability can be obtained from the phrase tables generated during training. The weighted entropy measure $d_{sce_{wt}}$,for each language, was normalised with the total number of such n-gram pairs being considered for comparison with other languages. We have fixed the maximum order of l and f n-grams to be 6. Table 3 shows the difficulty levels as calculated using WSCE along with the accuracy for the languages that we tested on. As is evident from this table, there is a rough correlation between the difficulty level and the accuracy obtained, which also seems intuitively valid, given the nature of these languages and their orthographies.

Language	Datasets	$d_{sce_{wt}}$	Accuracy
English	CMUDict	0.30	63.81±0.47
French	Brulex	0.41	86.71±0.52
Dutch	Celex	0.45	91.63±0.24
German	Celex	0.49	90.20±0.25

Table 3: $d_{sce_{wt}}$ values predict the accuracy rates.

6 Error Analysis

In this section we present a summary of the error analysis for the output generated. We tried to observe if there exist any patterns in the words that were transcribed incorrectly.

The majority of errors occurred in the case of vowel transcription, and diphthong transcription in particular. In the case of English, this can be attributed to the phenomenon of lexical borrowing from a variety of sources as a result of which the number of sparse alignments is very high. The system is also unable to learn allophonic variation of certain kinds of consonantal phonemes, most notably fricatives like /s/ and /z/. This problem is exacerbated by the irregularity of allophonic variation in the language itself.

7 Conclusion and Future Work

In this paper we have tried to address the problem of letter-to-phoneme conversion by modeling it as an SMT problem and we have used minimum error rate training to obtain the suitable model parameters, which according to our knowledge, is a novel approach to L2P task. The results obtained are comparable to the state of the art system and our error analysis shows that a lot of improvement is still possible.

Intuitively, the performance of the system can be improved in at least two areas. First is the Minimum Error Rate Training (MERT) and the second is the decoding phase. Using phonetic feature based edit distance or string similarity as the loss function in the MERT implementation can improve results significantly. In addition, incorporating more model parameters and extensive testing of these parameters might improve the results of the system. We also plan to introduce a decoding scheme similar to the substring based transducer (Sherif and Kondrak, 2007) to improve the usage of lower order language models.

Acknowledgements

This work was supported by ILMT grant 11(10)/2006-HCC(TDIL).

References

Susan Bartlett, Grzegorz Kondrak, and Colin Cherry. 2008. Automatic syllabification with structured SVMs for letter-to-phoneme conversion. In *Proceedings of ACL-08: HLT*, pages 568–576, Columbus, Ohio, June. ACL.

Max Bisani and Hermann Ney. 2002. Investigations on joint-multigram models for grapheme-to-phoneme conversion. In *International Conference on Spoken Language Processing*, pages 105–108, Denver, CO, USA, September.

A.W. Black, K. Lenzo, and V. Pagel. 1998. Issues in Building General Letter to Sound Rules. In *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*. ISCA.

M. Collins. 2002. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on EMNLP-Volume 10*, pages 1–8. ACL, Morristown, NJ, USA.

K. Crammer and Y. Singer. 2003. Ultraconservative online algorithms for multiclass problems. *The Journal of Machine Learning Research*, 3:951–991.

Walter M. P. Daelemans and Antal P. J. van den Bosch. 1997. Language-Independent Data-0riented Grapheme-to-Phoneme Conversion. *Progress in Speech Synthesis*.

R.I. Damper, Y. Marchand, J.D. Marseters, and A. Bazin. 2004. Aligning Letters and Phonemes for Speech Synthesis. In *Fifth ISCA Workshop on Speech Synthesis*. ISCA.

Sittichai Jiampojamarn, Grzegorz Kondrak, and Tarek Sherif. 2007. Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. In *HLT 2007: The Conference of the NAACL; Proceedings of the Main Conference*, pages 372–379, Rochester, New York, April. ACL.

Sittichai Jiampojamarn, Colin Cherry, and Grzegorz Kondrak. 2008. Joint processing and discriminative training for letter-to-phoneme conversion. In *Proceedings of ACL-08: HLT*, pages 905–913, Columbus, Ohio, June. ACL.

P. Koehn, F.J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the NAACL:HLT-Volume 1*, pages 48–54. ACL Morristown, NJ, USA.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL*, volume 45, page 2.

J. Kominek and A.W. Black. 2006. Learning pronunciation dictionaries: language complexity and word selection strategies. In *HLT-NAACL*, pages 232–239. ACL, Morristown, NJ, USA.

Y. Marchand and R.I. Damper. 2000. A Multistrategy Approach to Improving Pronunciation by Analogy. *Computational Linguistics*, 26(2):195–219.

F.J. Och and H. Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

F.J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on ACL-Volume 1*, pages 160–167. ACL, Morristown, NJ, USA.

J. Schroeter, A. Conkie, A. Syrdal, M. Beutnagel, M. Jilka, V. Strom, Y.J. Kim, H.G. Kang, and D. Kapilow. 2002. A Perspective on the Next Challenges for TTS Research. In *IEEE 2002 Workshop on Speech Synthesis*.

Tarek Sherif and Grzegorz Kondrak. 2007. Substring-based transliteration. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 944–951, Prague, Czech Republic, June. Association for Computational Linguistics.

A. Stolcke. 2002. Srilm – an extensible language modeling toolkit.

P. Taylor. 2005. Hidden Markov Models for Grapheme to Phoneme Conversion. In *Ninth European Conference on Speech Communication and Technology*. ISCA.

K. Toutanova and R.C. Moore. 2002. Pronunciation modeling for improved spelling correction. In *Proceedings of the 40th annual meeting of ACL*, pages 144–151.

A. van den Bosch and S. Canisius. 2006. Improved morpho-phonological sequence processing with constraint satisfaction inference. In *Proceedings of the Eighth Meeting of the ACL-SIGPHON at HLT-NAACL*, pages 41–49.

A. van den Bosch and W. Daelemans. 1998. Do not forget: Full memory in memory-based learning of word pronunciation. *proceedings of NeMLap3/CoNLL98*, pages 195–204.

R. Zens and H. Ney. 2004. Improvements in phrase-based statistical machine translation. In *HLT Conf. / NAACL*, pages 257–264, Boston, MA, May.

Disambiguation of Preposition Sense Using Linguistically Motivated Features

Stephen Tratz and Dirk Hovy
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292
{stratz,dirkh}@isi.edu

Abstract

In this paper, we present a supervised classification approach for disambiguation of preposition senses. We use the SemEval 2007 Preposition Sense Disambiguation datasets to evaluate our system and compare its results to those of the systems participating in the workshop. We derived linguistically motivated features from both sides of the preposition. Instead of restricting these to a fixed window size, we utilized the phrase structure. Testing with five different classifiers, we can report an increased accuracy that outperforms the best system in the SemEval task.

1 Introduction

Classifying instances of polysemous words into their proper sense classes (aka sense disambiguation) is potentially useful to any NLP application that needs to extract information from text or build a semantic representation of the textual information. However, to date, disambiguation between preposition senses has not been an object of great study. Instead, most word sense disambiguation work has focused upon classifying noun and verb instances into their appropriate WordNet (Fellbaum, 1998) senses. Prepositions have mostly been studied in the context of verb complements (Litkowski and Hargraves, 2007). Like instances of other word classes, many prepositions are ambiguous, carrying different semantic meanings (including notions of instrumental, accompaniment, location, etc.) as in "He ran with determination", "He ran with a broken leg", or "He ran with Jane". As NLP systems take more and more semantic content into account, disambiguating between preposition senses becomes increasingly important for text processing tasks.

In order to disambiguate different senses, most systems to date use a fixed window size to derive classification features. These may or may not be syntactically related to the preposition in question, resulting–in the worst case–in an arbitrary bag of words. In our approach, we make use of the phrase structure to extract words that have a certain syntactic relation with the preposition. From the words collected that way, we derive higher level features.

In 2007, the SemEval workshop presented participants with a formal preposition sense disambiguation task to encourage the development of systems for the disambiguation of preposition senses (Litkowski and Hargraves, 2007). The training and test data sets used for SemEval have been released to the general public, and we used these data to train and test our system. The SemEval workshop data consists of instances of 34 prepositions in natural text that have been tagged with the appropriate sense from the list of the common English preposition senses compiled by The Preposition Project, cf. Litkowski (2005). The SemEval data provides a natural method for comparing the performance of preposition sense disambiguation systems. In our paper, we follow the task requirements and can thus directly compare our results to the ones from the study. For evaluation, we compared our results to those of the three systems that participated in the task (MELB: Ye and Baldwin (2007); KU: Yuret (2007); IRST: Popescu et al. (2007)). We also used the "first sense" and the "most frequent sense"

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 96–100,
Boulder, Colorado, June 2009.

baselines (see section 3 and table 1). These baselines are determined by the TPP listing and the frequency in the training data, respectively. Our system beat the baselines and outperformed the three participating systems.

2 Methodology

2.1 Data Preparation

We downloaded the test and training data provided by the SemEval-2007 website for the preposition sense disambiguation task. These are 34 separate XML files–one for each preposition–, comprising 16557 training and 8096 test example sentences, each sentence containing one example of the respective preposition.

What are your beliefs <head>about</head> these emotions ?

The preposition is annotated by a head tag, and the meaning of the preposition in question is given as defined by TPP.

Each preposition had between 2 and 25 different senses (on average 9.76). For the case of "about" these would be

1. on the subject of; concerning
2. so as to affect
3. used to indicate movement within a particular area
4. around
5. used to express location in a particular place
6. used to describe a quality apparent in a person

We parsed the sentences using the Charniak parser (Charniak, 2000). Note that the Charniak parser–even though among the best availbale English parsers–occasionally fails to parse a sentence correctly. This might result in an erroneous extraction, such as an incorrect or no word. However, these cases are fairly rare, and we did not manually correct this, but rather relied on the size of the data to compensate for such an error.

After this preprocessing step, we were able to extract the features.

2.2 Feature Extraction

Following O'Hara and Wiebe (2003) and Alam (2004), we assumed that there is a meaningful connection between syntactically related words on both sides of the preposition. We thus focused on specific words that are syntactically related to the preposition via the phrase structure. This has the advantage that it is not limited to a certain window size; phrases might stretch over dozens of words, so the extracted word may occur far away from the actual preposition. These words were chosen based on a manual analysis of training data. Using Tregex (Levy and Andrew, 2006), a utility for expressing "regular expressions over trees", we created a set of rules to extract the words in question. Each rule matched words that exhibited a specific relationship with the preposition or were within a two word window to cover collocations. An example rule is given below.

$$IN > (PP < (VP < \#\_\_ = x \& < \#!AUX))$$

This particular rule finds the head (denoted by x) of a verb phrase that governs the prepositional phrase containing the preposition, unless x is an auxiliary verb. Tregex rules were used to identify the following words for feature generation:

- the head verb/noun that immediately dominates the preposition along with all of its modifying determiners, quantifiers, numbers, and adjectives
- the head verb/noun immediately dominated by the preposition along with all of its modifying determiners, quantifiers, numbers, and adjectives
- the subject, negator, and object(s) of the immediately dominating verb
- neighboring prepositional phrases dominated by the same verb/noun ("sister" prepositional phrases)
- words within 2 positions to the left or right of the preposition

For each word extracted using these rules, we collected the following items:

- the word itself
- lemma
- part-of-speech (both exact and conflated, e.g. both 'VBD' and 'verb' for 'VBD')
- all synonyms of the first WordNet sense
- all hypernyms of the first WordNet sense
- boolean indicator for capitalization

Each feature is a combination of the extraction rule and the extracted item. The values the feature can take on are binary: present or absent. For some prepositions, this resulted in several thousand features. In order to reduce computation time, we used the following steps: For each preposition classifier, we ranked the features using information gain (Forman, 2003). From the resulting lists,we included at most 4000 features. Thus not all classifiers used the same features.

2.3 Classifier Training

We chose maximum entropy (Berger et al., 1996) as our primary classifier, since it had been successfully applied by the highest performing systems in both the SemEval-2007 preposition sense disambiguation task (Ye and Baldwin, 2007) and the general word sense disambiguation task (Tratz et al., 2007). We used the implementation provided by the Mallet machine learning toolkit (McCallum, 2002). For the sake of comparison, we also built several other classifiers, including multinomial naïve Bayes, SVMs, kNN, and decision trees (J48) using the WEKA toolkit (Witten, 1999). We chose the radial basis function (RBF) kernel for the SVMs and left all other parameters at their default values.

3 Results

We measured the accuracy of the classifiers over the test set provided by SemEval-2007 and provided these results in Table 1. It is notable that our system produced good results with all classifiers: For three of the classifiers, the accuracy is higher than MELB, the winning system of the task. As expected, the highest accuracy was achieved using the maximum entropy classifier. Overall, our system outperformed the winning system by 0.058, an 8 percent improvement. A simple proportion test shows this to be statistically significant at 0.001.

System	Accuracy
kNN	0.0684
SVM (RBF kernel)	0.0692
J48 decision trees	0.0712
Multinomial Naïve Bayes	0.0731
Maximum entropy	0.0751
MELB (Ye and Baldwin, 2007)	0.0693
KU (Yuret, 2007)	0.0547
IRST (Popescu et al., 2007)	0.0496
Most frequent sense	0.0396

Table 1: Accuracy results on SemEval data (with 4000 features)

Since our initial cutoff of 4000 features was arbitrary, we reran our Maximum Entropy experiment multiple times with different cutoffs. Accuracy consistently increased as the feature limit was relaxed, resulting in 0.764 accuracy at the 10k feature limit. These results are displayed in Figure 1.

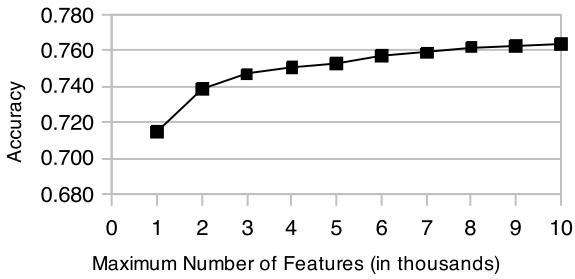

Figure 1: Maximum feature limit vs. accuracy for maximum entropy classifier

4 Related Work

The linguistic literature on prepositions and their use is copious and diverse. We restrict ourselves to the systems that competed in the SemEval 2007 Preposition Sense Disambiguation task. All three of the systems within the framework of the SemEval task used supervised learning algorithms, yet they differed widely in the data collection and model preparation.

Ye and Baldwin (2007) participated in the SemEval task using a maximum entropy classifier and achieved the highest accuracy of the participating systems. The features they extracted were similar to the ones we used, including POS and WordNet features, but they used a substantially larger word window, taking seven words from each side of the preposition. While they included many higher level features, they state that the direct lexical context (i.e., bag-of-words) features were the most effective and account for the majority of features, while syntactic and semantic features had relatively little impact.

Yuret (2007) used a n-gram model based on word substitution by synonyms or antonyms. While this proved to be quite successful with content words, it had considerable problems with prepositions, since the number of synonyms and/or antonyms is fairly limited.

Popescu et al. (2007) take an interesting approach which they call Chain Clarifying Relationship. They are using a supervised algorithm to learn a regular language. They used the Charniak parser and FrameNet information on the head, yet the features they extract are generally not linguistically motivated.

5 Discussion

Using the phrase structure allows for more freedom in the choice of words for feature selection, yet still guarantees to find words for which some syntactic relation with the preposition holds. Extracting semantic features from these words (hypernyms, synonyms, etc.) allows for a certain degree of abstraction, and thus a high level comparison. O'Hara and Wiebe (2003) also make use of high level features, in their case the Penn Treebank (Marcus et al., 1993) and FrameNet (Baker et al., 1998) to classify prepositions. They show that using high level features–such as semantic roles–of words in the context substantially aids disambiguation efforts. They caution, however, that indiscriminately using collocations and neighboring words may yield high accuracy, but has the risk of overfitting. In order to mitigate this, they classify the features by their part of speech. While we made use of collocation features, we also took into account higher order aspects of the context, such as the governing phrase, part of speech type, and semantic class according to WordNet. All other things being equal, this seems to increase performance substantially.

As for the classifiers used, our results seem to confirm that Maximum Entropy classifiers are very well suited for disambiguation tasks. Other than naïve Bayes, they do not presuppose a conditional independence between the features, which clearly not always holds (quite contrary, the underlying syntactic structure creates strong interdependencies between words and features). This, however, does not satisfactory explain the ranking of the other classifiers. One possible explanation could be the sensitivity of for example decision trees to random noise. Though we made use of information gain before classification, there still seems to be a certain tendency to split on features that are not optimal.

6 Conclusion

We showed that using a number of simple linguistically motivated features can improve the accuracy of preposition sense disambiguation. Utilizing widely used and freely available standard tools for language processing and a set of simple rules, we were able to extract these features easily and with very limited preprocessing. Instead of taking a "bag of words" approach that focuses primarily upon the words within a fixed window size, we focused on elements that are related via the phrase structure. We also included semantic information gathered from WordNet about the extracted words. We compared five different classifiers and demonstrated that they all perform very well, using our selected feature set. Several of them even outperformed the top system at SemEval. Our best result was obtained using a maximum entropy classifier, just as the best participating system, leading us to believe that our primary advantage was our feature set. While the contribution of the direct context (+/-7 words) might have a stronger effect than higher level features (Ye and Baldwin, 2007), we conclude from our findings that higher level features do make an important contribution. These results are very encouraging on several levels, and demonstrate the close interaction of syntax and semantics. Leveraging these types of features effectively is a promising prospect for future

machine learning research in preposition sense disambiguation.

Acknowledgements

The authors would like to thank Eduard Hovy and Gully Burns for invaluable comments and helpful discussions.

References

Y.S. Alam. 2004. Decision Trees for Sense Disambiguation of Prepositions: Case of Over. In *HLT-NAACL 2004: Workshop on Computational Lexical Semantics*, pages 52–59.

C.F. Baker, C.J. Fillmore, and J.B. Lowe. 1998. The Berkeley FrameNet Project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics Morristown, NJ, USA.

A.L. Berger, V.J. Della Pietra, and S.A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

E. Charniak. 2000. A maximum-entropy-inspired parser. In *ACM International Conference Proceeding Series*, volume 4, pages 132–139.

C. Fellbaum. 1998. *WordNet: an electronic lexical database*. MIT Press USA.

G. Forman. 2003. An extensive empirical study of feature selection metrics for text classification. *The Journal of Machine Learning Research*, 3:1289–1305.

R. Levy and G. Andrew. 2006. Tregex and Tsurgeon: tools for querying and manipulating tree data structures. In *LREC 2006*.

Ken Litkowski and Orin Hargraves. 2007. SemEval-2007 Task 06: Word-Sense Disambiguation of Prepositions. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, Prague, Czech Republic.

Ken Litkowski. 2005. The preposition project. http://www.clres.com/prepositions.html.

M.P. Marcus, M.A. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of English: the Penn TreeBank. *Computational Linguistics*, 19(2):313–330.

A.K. McCallum. 2002. MALLET: A Machine Learning for Language Toolkit. 2002. http://mallet. cs. umass. edu.

T. O'Hara and J. Wiebe. 2003. Preposition semantic classification via Penn Treebank and FrameNet. In *Proceedings of CoNLL*, pages 79–86.

Octavian Popescu, Sara Tonelli, and Emanuele Pianta. 2007. IRST-BP: Preposition Disambiguation based on Chain Clarifying Relationships Contexts. In *MELB-YB: Preposition Sense Disambiguation Using Rich Semantic Features*, Prague, Czech Republic.

S. Tratz, A. Sanfilippo, M. Gregory, A. Chappell, C. Posse, and P. Whitney. 2007. PNNL: A Supervised Maximum Entropy Approach to Word Sense Disambiguation. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*.

I.H. Witten. 1999. *Weka: Practical Machine Learning Tools and Techniques with Java Implementations*. Dept. of Computer Science, University of Waikato, University of Waikato, Dept. of Computer Science.

Patrick Ye and Timothy Baldwin. 2007. MELB-YB: Preposition Sense Disambiguation Using Rich Semantic Features. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, Prague, Czech Republic.

Deniz Yuret. 2007. Ku: Word sense disambiguation by substitution. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, Prague, Czech Republic.

Author Index