NAACL HLT 2007

Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics

Companion Volume: Short Papers

Candace Sidner, General Chair
Tanja Schultz, Matthew Stone, and ChengXiang Zhai
Program Committee Chairs

22–27 April 2007
Rochester, New York, USA

Production and Manufacturing by
Omnipress Inc.
2600 Anderson Street
Madison, WI 53704

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: +1-507-476-0860
acl@aclweb.org

Table of Contents

Comparing User Simulation Models For Dialog Strategy Learning

Hua Ai
University of Pittsburgh
Intelligent Systems Program
Pittsburgh PA, 15260, USA
hua@cs.pitt.edu

Joel R. Tetreault
University of Pittsburgh
LRDC
Pittsburgh PA, 15260, USA
tetreaul@pitt.edu

Diane J. Litman
University of Pittsburgh
Dept. of Computer Science
LRDC
Pittsburgh PA, 15260, USA
litman@cs.pitt.edu

Abstract

This paper explores what kind of user simulation model is suitable for developing a training corpus for using Markov Decision Processes (**MDPs**) to automatically learn dialog strategies. Our results suggest that with sparse training data, a model that aims to randomly explore more dialog state spaces with certain constraints actually performs at the same or better than a more complex model that simulates realistic user behaviors in a statistical way.

1 Introduction

Recently, user simulation has been used in the development of spoken dialog systems. In contrast to experiments with human subjects, which are usually expensive and time consuming, user simulation generates a large corpus of user behaviors in a low-cost and time-efficient manner. For example, user simulation has been used in evaluation of spoken dialog systems (López-Cózar et al., 2003) and to learn dialog strategies (Scheffler, 2002). However, these studies do not systematically evaluate how helpful a user simulation is. (Schatzmann et al., 2005) propose a set of evaluation measures to assess the realness of the simulated corpora (i.e. how similar are the simulated behaviors and human behaviors). Nevertheless, how realistic a simulated corpus needs to be for different tasks is still an open question.

We hypothesize that for tasks like system evaluation, a more realistic simulated corpus is preferable. Since the system strategies are evaluated and adapted based on the analysis of these simulated dialog behaviors, we would expect that these behaviors are what we are going to see in the test phase when the systems interact with human users. However, for automatically learning dialog strategies, it is not clear how realistic versus how exploratory (Singh et al., 2002) the training corpus should be. A training corpus needs to be exploratory with respect to the chosen dialog system actions because if a certain action is never tried at certain states, we will not know the value of taking that action in that state. In (Singh et al., 2002), their system is designed to randomly choose one from the allowed actions with uniform probability in the training phase in order to explore possible dialog state spaces. In contrast,we use user simulation to generate exploratory training data because in the tutoring system we work with, reasonable tutor actions are largely restricted by student performance. If certain student actions do not appear, this system would not be able to explore a state space randomly .

This paper investigates what kind of user simulation is good for using Markov Decision Processes (**MDPs**) to learn dialog strategies. In this study, we compare three simulation models which differ in their efforts on modeling the dialog behaviors in a training corpus versus exploring a potentially larger dialog space. In addition, we look into the impact of different state space representations and different reward functions on the choice of simulation models.

2 System and Corpus

Our system is a speech-enabled Intelligent Tutoring System that helps students understand qualita-

Proceedings of NAACL HLT 2007, Companion Volume, pages 1–4,
Rochester, NY, April 2007.

tive physics questions. The dialog policy was deterministic and hand-crafted in a finite state paradigm (Ai et al., 2006). We collected 130 dialogs (1019 student utterances) with 26 human subjects. Correctness (correct(**c**), incorrect(**ic**)) is automatically judged by the system[1] and kept in the system's logs. Percent incorrectness (**ic%**) is also automatically calculated and logged. Each student utterance was manually annotated for certainty (*certain*, *uncertain*, *neutral*, *mixed*) in a previous study[2] based on both lexical and prosodic information. In this study, we use a two-way classification (certain(**cert**), not-certain(**ncert**)), where we collapse *uncertain*, *neutral*, and *mixed* to be **ncert** to balance our data. An example of coded dialog between the tutor (**T**) and a student (**S**) is given in Table 1.

T1:	Which law of motion would you use?
S1:	Newton's second law? [**ic**, **ic%**=1, **ncert**]
T2:	Well... The best law to use is Newton's third law. Do you recall what it says?
S2:	For every action there is an equal and opposite reaction? [**c**, **ic%**=50%, **ncert**]

Table 1: Sample coded dialog excerpt.

3 Experimental Setup

3.1 Learning Task

Our current system can only respond to the correctness of a student's utterances; the system thus ignores other underlying information, for example, certainty which is believed to provide useful information for the tutor. In our corpus, the strength of the tutor's minimal feedback (defined below) is in fact strongly correlated with the percentage of student certainty (chi-square test, $p<0.01$). Strong Feedback (**SF**) is when the tutor clearly states whether the student's answer is correct or incorrect (i.e., "This is great!"); Weak Feedback (**WF**) is when the tutor does not comment on the correctness of a student's answer or gives slightly negative feedback such as "well". Our goal is to learn how to manipulate the strength of the tutor minimal feedback in order to maximize student's overall certainty in the entire dialog. We keep the other parts of the tutor feedback (e.g. explanations, questions) so the system's original design of maximizing the percentage of student correct answers is utilized.

3.2 Simulation Models

All three models we describe below are trained from the real corpus we collected. We simulate on the word level because generating student's dialog acts alone does not provide sufficient information for our tutoring system to decide the next system's action. Thus, the output of the three models is a student utterance along with the student certainty (**cert**, **ncert**). Since it is hard to generate a natural language utterance for each tutor's question, we use the student answers in the real corpus as the candidate answers for the simulated students (Ai et al., 2006). In addition, we simulate student certainty in a very simple way: the simulation models output the certainty originally associated with that utterance.

Probabilistic Model (PM) is meant to capture realistic student behavior in a probabilistic way. Given a certain tutor question along with a tutor feedback, it will first compute the probabilities of the four types of student answers from the training corpus: **c** and **cert**, **c** and **ncert**, **ic** and **cert**, and **ic** and **ncert**. Then, following this distribution, the model selects the type of student answers to output, and then it picks an utterance that satisfies the correctness and certainty constraints of the chosen answer type from the candidate answer set and outputs that utterance. We implement a back-off mechanism to count possible answers that do not appear in the real corpus.

Total Random Model (TRM) ignores what the current question is or what feedback is given. It randomly picks one utterance from all the utterances in the entire candidate answer set. This model tries to explore all the possible dialog states.

Restricted Random Model (RRM) differs from the **PM** in that given a certain tutor question and a tutor feedback, it chooses to give a **c** and **cert**, **c** and **ncert**, **ic** and **cert**, or **ic** and **ncert** answer with equal probability. This model is a compromise between the exploration of the dialog state space and the realness of generated user behaviors.

3.3 MDP Configuration

A MDP has four main components: states, actions, a policy, and a reward function. In this study, the actions allowed in each dialog state are **SF** and **WF**;

[1] Kappa of 0.79 is gained comparing to human judgements.
[2] Kappa of 0.68 is gained in a preliminary agreement study.

the policy we are trying to learn is in every state whether the tutor should give **SF** and **WF** in order to maximize the percent certainty in the dialog.

Since different state space representations and reward functions have a strong impact on the MDP policy learning, we investigate different configurations to avoid possible bias introduced by certain configurations. We use two state space representations: **SSR1** uses the correctness of current student turn and percent incorrectness so far; and **SSR2** adds in the certainty of the current student turn on top of **SSR1**. Two reward functions are investigated: in **RF1**, we assign +100 to every dialog that has a percent certainty higher than the median from the training corpus, and -100 to every dialog that has a percent certainty below the median; in **RF2**, we assign different rewards to every different dialog by multiplying the percent certainty in that dialog with 100. Other MDP parameter settings are the same as described in (Tetreault et al., 2006).

3.4 Methodology

We first let the three simulation models interact with the original system to generate different training corpora. Then, we learn three MDP policies in a fixed configuration from the three training corpora separately. For each configuration, we run the simulation models until we get enough training data such that the learned policies on that corpus do not change anymore (40,000 dialogs are generated by each model). After that, the learned new policies are implemented into the original system respectively [3]. Finally, we use our most realistic model, the **PM**, to interact with each new system 500 times to evaluate the new systems' performances. We use two evaluation measures. **EM1** is the number of dialogs that would be assigned +100 using the old median split. **EM2** is the average of percent certainty in every single dialog from the newly generated corpus. A policy is considered better if it can improve the percentage of certainty more than other policies, or has more dialogs that will be assigned +100. The baseline for **EM1** is 250, since half of the 500 dialogs would be assigned +100 using the old median split. The baseline for **EM2** is 35.21%, which is obtained by calculating the percent certainty in the corpus generated by the 40,000 interactions between the **PM** and the original system.

4 Results and Discussion

Table 2 summarizes our results. There are two columns under each "state representation+reward function" configuration, presenting the results using the two evaluation approaches. **EM1** measures exactly what **RF1** tries to optimize; while **EM2** measures exactly what **RF2** tries to optimize. However, we show the results evaluated by both **EM1** and **EM2** for all configurations since the two evaluation measures have their own practical values and can be deployed under different design requirements. All results that significantly[4] outperform the corresponding baselines are marked with $*$.

When evaluating using **EM1**, the **RRM** significantly[4] outperforms the other two models in all configurations (in **bold** in Table 2). Also, the **PM** performs better (but not statistically significantly) than the **TRM**. When evaluating on **EM2**, the **RRM** significantly[4] outperforms the other two when using **SSR1** and **RF1** (in **bold** in Table 2). In all other configurations, the three models do not differ significantly. It is not surprising that the **RRM** outperforms the **PM** in most of the cases even when we test on the **PM**. (Schatzmann et al., 2005) also observe that a good model can still perform well when tested on a poor model.

We suspect that the performance of the **PM** is harmed by the data sparsity issue in the real corpus that we trained the model on. Consider the case of **SSR1**: 25.8% of the potentially possible dialog states do not exist in the real corpus. Although we implement a back-off mechanism, the **PM** will still have much less chance to transition to the states that are not observed in the real corpus. Thus, when we learn the MDP policy from the corpus generated by this model, the actions to take in these less-likely states are not fully learned. In contrast, the **RRM** transitions from one state to each of the next possible states with equal probability, which compensates for the data sparsity problem. We further examine the results obtained using **SSR1** and **RF1** and evaluated

[3] For example, the policy learned from the training corpus generated by the **RRM** with **SSR1** and **RF1** is: give **SF** when the current student answer is **ic** and **ic%**$>$50%, otherwise give **WF**.

[4] Using 2-sided t-test with Bonferroni correction, $p<0.05$.

Model Name	SSR1+RF1		SSR2+RF1		SSR1+RF2		SSR2+RF2	
	EM1	EM2	EM1	EM2	EM1	EM2	EM1	EM2
Probabilistic Model	222	36.30%	217	37.63%	197	40.78%∗	197	40.01%∗
Total Random Model	192	36.30%	211	38.57%	188	40.21%∗	179	40.21%∗
Restricted Random Model	**390**∗	**46.11%**∗	**368**∗	37.27%	**309**	40.21%∗	**301**	40.21%∗

Table 2: Evaluation of the new policies trained with the three simulation models

by **EM1** to confirm our hypothesis. When looking into the frequent states[5], 70.1% of them are seen frequently in the training corpus generated by the **PM**, while 76.3% are seen frequently in the training corpus generated by the **RRM**. A higher percentage indicates the policy might be better trained with more training instances. This explains why the **RRM** outperforms the **PM** in this case.

While the **TRM** also tries to explore dialog state space, only 65.2% of the frequent states in testing phase are observed frequently in the training phase. This is because the Total Random Model answers 90% of the questions incorrectly and often goes deeply down the error-correction paths. It does explore some states that are at the end of the paths, but since these are the infrequent states in the test phase, exploring these states does not actually improve the model's performance much. On the other hand, while the student correctness rate in the real corpus is 60%, the **RRM** prevents itself from being trapped in the less-likely states on incorrect answer paths by keeping its correctness rate to be 50%.

Our results are preliminary but suggest interesting points in building simulation models: 1. When trained from a sparse data set, it may be better to use a **RRM** than a more realistic **PM** or a more exploratory **TRM**; 2. State space representation may not impact evaluation results as much as reward functions and evaluation measures, since when using **RF2** and evaluating with **EM2**, the differences we see using **RF1** or **EM1** become less significant.

In our future work, we are going to further investigate whether the trends shown in this paper generalize to on-line MDP policy learning. We also want to explore other user simulations that are designed for sparse training data (Henderson et al., 2005). More importantly, we are going to test the new policies with the other simulations and human subjects to validate the learning process.

[5]We define frequent states to be those that comprise at least 1% of the entire corpus. These frequent states add up to more than 80% of the training/testing corpus. However, deciding the threshold of the frequent states in training/testing is an open question.

Acknowledgements

NSF (0325054, 0328431) supports this research. The authors wish to thank Tomas Singliar for his valuable suggestions, Scott Silliman for his support on building the simulation system, and the anonymous reviewers for their insightful comments.

References

H. Ai and D. Litman. 2006. *Comparing Real-Real, Simulated-Simulated, and Simulated-Real Spoken Dialogue Corpora*. In Proc. AAAI Workshop on Statistical and Empirical Approaches for SDS.

J. Henderson, O.Lemon, and K.Georgila. 2005. *Hybrid reinforcement/supervised learning for dialogue policies from COMMUNICATOR data*. In Proc. IJCAI workshop on Knowledge and Reasoning in Practical Dialogue Systems.

R. López-Cózar, A. De la Torre, J. Segura, and A. Rubio. 2003. *Assessment of dialog systems by means of a new simulation technique*. Speech Communication (40): 387-407.

K. Scheffler. 2002. *Automatic Design of Spoken Dialog Systems*. Ph.D. diss., Cambridge University.

J. Schatzmann, K. Georgila, and S. Young. 2005. *Quantitative Evaluation of User Simulation Techniques for Spoken Dialog Systems*. In Proc. of 6th SIGdial.

J. Schatzmann, M. N. Stuttle, K. Weilhammer and S. Young. 2005. *Effects of the User Model on Simulation-based Learning of Dialogue Strategies*. In Proc. of ASRU05.

S. Singh, D. Litman, M. Kearns, and M. Walker. 2002. *Optimizing Dialog Managment with Reinforcement Learning: Experiments with the NJFun System*. Journal of Artificial Intelligence Research, (16):105-133.

J. Tetreault and D. Litman. 2006. *Comparing the Utility of State Features in Spoken Dialogue Using Reinforcement Learning.*. In Proc. NAACL06.

Automatic acquisition of grammatical types for nouns

Núria Bel Sergio Espeja Montserrat Marimon

IULA
Universitat Pompeu Fabra
P. de la Mercè, 10-12
ES-08002 – Barcelona
{nuria.bel,sergio.espeja,montserrat.marimon}@upf.edu

Abstract

The work[1] we present here is concerned with the acquisition of deep grammatical information for nouns in Spanish. The aim is to build a learner that can handle noise, but, more interestingly, that is able to overcome the problem of sparse data, especially important in the case of nouns. We have based our work on two main points. Firstly, we have used distributional evidences as features. Secondly, we made the learner deal with all occurrences of a word as a single complex unit. The obtained results show that grammatical features of nouns is a level of generalization that can be successfully approached with a Decision Tree learner.

1 Introduction

Our work aims to the acquisition of deep grammatical information for nouns, because having information such as countability and complementation is necessary for different applications, especially for deep analysis grammars, but also for question answering, topic detection and tracking, etc.

Most successful systems of deep lexical acquisition are based on the idea that distributional features (i.e. the contexts where words occur) are associated to concrete lexical types. The difficulties are, on the one hand, that some filtering must be applied to get rid of noise, that is, contexts wrongly assessed as cues of a given type and, on the other hand, that for a pretty large number of words, their occurrences in a corpus of any length are very few, making statistical treatment very difficult.

The phenomenon of noise is related to the fact that one particular context can be a cue of different lexical types. The problem of sparse data is predicted by the Zipfian distribution of words in texts: there is a large number of words likely to occur a very reduced number of times in any corpus. Both of these typical problems are maximized in the case of nouns.

The aim of the work we present here is to build a learner that can handle noise, but, more interestingly, that is able to overcome the problem of sparse data. The learner must predict the correct type both when there is a large number of occurrences as well as when there are only few occurrences, by learning on features that maximize generalization capacities of the learner while controlling overfitting phenomena.

We have based our work on two main points. Firstly, we have used morphosyntactic information as features. Secondly, we made the learner deal with all occurrences of a word as a complex unit. In our system, linguistic cues of every occurrence are collected in the *signature* of the word (more technically a pair *lema* + *part of speech*) in a particular corpus. In the next sections we give further details about the features used, as well as about the use of signatures.

The rest of the paper is as follows. Section 2 presents an overview of the state of the art in deep lexical acquisition. In section 3, we introduce details about our selection of linguistically motivated

[1] This research was supported by the Spanish Ministerio de Educación y Ciencia: project AAILE, HUM2004-05111-C02-01/FILO, Ramón y Cajal, Juan de la Cierva Programs and PTA-CTE/1370/2003 with *Fondo Social Europeo*,.

Proceedings of NAACL HLT 2007, Companion Volume, pages 5–8,
Rochester, NY, April 2007.

cues to be used as features for training a Decision Tree (DT). Section 4 shortly introduces the methodology and data used in the experiments whose results are presented in section 5. And in section 6 we conclude by comparing with the published results for similar tasks and we sketch future research.

2 State of the art

Most of the work on deep lexical information acquisition has been devoted to verbs. The existing acquisition systems learn very specialized linguistic information such as verb subcategorization frame[2]. The results for verb subcategorization are mostly around the 0.8 of precision. Briscoe & Carroll (1997) reported a type precision of 0,76 and a type recall of 0.43. Their results were improved by the work of Korhonen (2002) with a type precision of 0.87 and a recall of 0.68 using external resources to filter noise. Shulte im Walde (2002) reports a precision of 0.65 and a recall of 0.58. Chesley & Salmon-Alt (2006) report a precision of 0.86 and a recall of 0.54 for verb subcategorization acquisition for French.

Lexical acquisition for nouns has been concerned mainly with ontological classes and has mainly worked on measuring semantic similarity on the basis of occurrence contexts. As for grammatical information, the work of Baldwin and Bond (2003) in acquisition of countability features for English nouns also tackles the very important problem of feature selection. Other work like Carroll and Fang's (2004) and Baldwin's (2005) have focused on grammatical information acquisition for HPSG based computational grammars. The latter is the most similar exercises to our work. Baldwin (2005) reports his better results in terms of type accuracy has been obtained by using syntactic information in a chunked and parsed corpus. The type F-scores for the different tested categories for English were: for verbs 0.47, for nouns 0.6 and for adjectives 0.832.

3 Feature selection

One of the most important tasks in developing machine learning applications is the selection of the features that leads to the smallest classification error. For our system, we have looked at distributional motivated features that can help in discriminating the different types that we ultimately use to classify words.

The lexical types used in deep analysis grammars are linguistic generalizations drawn from the distributional characteristics of particular sets of words. For the research we present here, we have taken the lexicon of a HPSG-based grammars developed in the LKB platform (Copestake, 2002) for Spanish, similarly to the work of Baldwin (2005). In the LKB grammatical framework, lexical types are defined as a combination of features. Lexical typology of nouns for Spanish, for instance, can be seen as a cross-classification of noun countability vs. mass distinctions, and subcategorization frame or valence, including prepositional selection. For example nouns as "temor" ('fear') and "adicción" ('adiction) belong to the type `n_ppde_pcomp_a_count` as they take two complements: one with *de* and the other with a bound preposition *a*, as in "El temor de la niña a los fantasmas" ('The girl's fear to ghosts') vs. "La adicción a la cocaína" ('The addiction to cocaine').

We decided to carry out the classification for each of the grammatical features that conform the cross-classified types as a better level of generalization than the type: *mass* and *countable*, on the one hand and, on the other hand, for subcategorization information three further basic features: *trans*, for nouns with thematic complements introduced by the preposition *de*, *intrans*, when the noun can appear with no complements and *pcomp* for nouns having complements introduced by a bound preposition. The complete type can be recomposed with the assigned features. "Temor" and "adicción" will be examples of *trans* and *pcomp_a*. They both have also to be assigned the feature *countable*. The combination of features assigned corresponds to the final type which is a definition of the complete behaviour of the noun with respect, for instance, optional complements.

We have used 23 linguistic cues, that is, the patterns of contexts that can be indicative of a particular feature. The most frequent cue that can be related to *countable* is for the noun to be found with plural morphology. A singular noun without determiner after a verb or a preposition is a cue of the noun being *mass*: "hay barro en el salón" ('there is mud in the living room") vs. "hay hombres en el

[2] Given the argument-adjunct distinction, subcategorization concerns the specification for a predicate of the number and type of arguments which it requires for well-formedness.

salón" ("there are men in the living room"). A further cue for *mass* is the presence of particular quantifiers, such as "más" ('more'), "menos" ('less'), etc. But these cues, based on a collection of lexical items, are less productive than other characteristics such as morphological number or presence of determiners, as they appear very scarcely in texts. Nevertheless, we should mention that most of mass nouns in Spanish can also appear in the contexts of countables, as in the case of "beer" when in constructions such as "three beers, please".

More difficult was to find cues for identifying the transitive nature of a noun. After some empirical work, we found a tendency of argumental complements to have a definite article: "temor de la niña" ('fear of the girl'), while modifiers tend to appear without determiners: "mesa de juegos" ('table of games'). Besides, we have taken as a cue the morphological characteristics of deverbal nouns. Suffixes such as "-ción", "-sión", and "-miento", are very much indicative of transitive nouns. Finally, to find the bound preposition of complements, we used a pattern for each possible preposition found after the noun in question.

We used Regular Expressions to implement the linguistic motivated patterns that check for the information just mentioned in a part of speech tagged corpus. The various patterns determine whether the linguistic cues that we have related to syntactic features are found in each occurrence of a particular word in a corpus. The positive or negative results of the *n* pattern checking are stored as binary values of a *n* dimensional vector, one for each occurrence. All vectors produced, one per occurrence of the word in question, are stored then in a kind of vector of vectors that we have called its *signature*. The term *signature* wants to capture the notion that the data it embodies is truly representative of a particular item, and that shows the details of its typical behavior. Particularly, we wanted linguistic cues appearing in different occurrences of the same word to be observed as related information. We have not dealt with ambiguity at all, however. One of the reasons was our focus on low frequency nouns.

4 Methodology and data

We have worked with the *Corpus Tècnic de l'IULA*, a multilingual part of speech tagged corpus which consists of domain specific texts. The section used for our evaluation was the Spanish with 1,091,314 words in the domain of economy and 4,301,096 for medicine. A dataset of 289 nouns, present in both subcorpora, was selected. It was important to compare the behavior of the same nouns in both corpus to check whether the learner was subject to unwanted overfitting.

We used the data for building a C4.5 DT classifier[3]. DT's are one well known and successful technique for this class of tasks when there is enough pre-annotated data available. DT's have the additional benefit that the results can be inspected. The signatures of the words in the Gold-Standard lists were extracted from the corpus of medicine and of the economy one. There was a further test set of 50 nouns with a single occurrence in the corpus of economy for testing purposes. The DT was trained with the signatures of the economy corpus, and the medicine ones as well as the singles set were used for testing.

5 Evaluation

The purpose of the evaluation was to validate our system with respect to the two problems mentioned: noise filtering and generalization capacity by measuring type precision and type recall. We understand type precision as a measure of the noise filtering success, and recall as a measure of the generalization capacity.

In the following tables we present the results of the different experiments. In Table 1, there is a view of the results of the experiment after training and testing with the signatures got in the smaller corpus. The results are for the assignment of the grammatical feature for the two values, *yes* and *no*. And the column named *global* refers to the total percentage of correctly classified instances.

		yes			*no*		
lt	***global***	***prec.***	***rec.***	***F***	***prec.***	***rec.***	***F***
MASS	0.67	0.4	0.26	0.31	0.73	0.83	0.78
COUNT	0.96	0.97	0.99	0.98	0	0	0
TRANS	0.85	0.73	0.45	0.55	0.86	0.95	0.91
INT	0.81	0.84	0.94	0.89	0.64	0.32	0.48
PCOMP	0.9	0.4	0.08	0.13	0.91	0.98	0.95

Table 1. DT results of economy signatures for training and test

3 We have used WEKA J48 decision tree classifier (Witten and Frank, 2005).

The most difficult task for the learner is to identify nouns with bound prepositions. Note that there are only 20 nouns with prepositional complements of the 289 test nouns, and that the occurrence of the preposition is not mandatory, and hence the signatures are presented to the learner with very little information.

Table 2 shows the results for 50 nouns with only one occurrence in the corpus. The performance does not change significantly, showing that the generalization capacity of the learner can cope with low frequency words, and that noise in larger signatures has been adequately filtered.

		yes			*no*		
lt	***global***	***prec.***	***rec.***	***F***	***prec.***	***rec.***	***F***
MASS	0.71	0.5	0.16	0.25	0.73	0.93	0.82
COUNT	0.97	0.97	1	0.98	0	0	0
TRANS	0.85	0.75	0.46	0.57	0.87	0.96	0.91
INT	0.83	0.85	0.95	0.89	0.70	0.41	0.52
PCOMP	0.91	0	0	0	0.91	1	0.95

Table 2. DT results for training with signatures of the economy corpus and testing 50 unseen nouns with a single occurrence as test

Table 3 shows that there is little variation in the results of training with signatures of the economy corpus and testing with ones of the medicine corpus. As expected, no variation due to domain is relevant as the information learnt should be valid in all domains.

		yes			*no*		
lt	***global***	***prec.***	***rec.***	***F***	***prec.***	***rec.***	***F***
MASS	0.65	0.44	0.53	0.48	0.77	0.70	0.73
COUNT	0.97	0.97	1	0.98	0	0	0
TRANS	0.82	0.62	0.47	0.54	0.86	0.92	0.89
INT	0.78	0.82	0.92	0.86	0.58	0.35	0.43
PCOMP	0.81	0.31	0.28	0.29	0.92	0.93	0.93

Table 3. DT results for training with economy signatures and testing with medicine signatures

6 Conclusions

The obtained results show that the learning of grammatical features of nouns are learned successfully when using distributional linguistic information as learning features that allow the learner to generalize so as to maintain the performance in cases of nouns with just one occurrence.

There are however issues that should be further investigated. Grammatical features with low precision and recall results (*mass* and *pcomp*) show that some more research should be carried out for finding relevant linguistic cues to be used as learning features. In that respect, the local cues based on morphosyntactic tagging have proved to be useful, minimizing the text preprocessing requirements for getting usable results.

Acknowledgements

The authors would like to thank Jordi Porta, Daniel Chicharro and the anonymous reviewers for helpful comments and suggestions.

References

Baldwin, T. 2005. "Bootstrapping Deep Lexical Resources: Resources for Courses", *ACL-SIGLEX 2005. Workshop on Deep Lexical Acquisition*.

Baldwin, T. and F. Bond. 2003. "Learning the Countability of English Nouns from Corpus Data". *Proceedings of the 41st. Annual Meeting of the ACL*.

Briscoe, T. and J. Carroll. 1997. "Automatic extraction of subcategorization from corpora". In *Proceedings of the Fifth Conference on Applied Natural Processing*, Washington.

Carroll, J. and A. Fang. 2004. "The automatic acquisition of verb subcategorisations and their impact on the performance of an HPSG parser". In *Proceedings of the 1st International Joint Conference on Natural Language Processing (IJCNLP)*, Sanya City, China.

Chesley, P and S. Salmon-Alt. 2006. "Automatic extraction of subcategorization frames for French". In *Proc. of the LREC Conference*, Genoa.

Copestake, A.. 2002. *Implementing Typed Feature Structure Grammars*. CSLI Publications.

Korhonen, A. 2002. "Subcategorization acquisition". As Technical Report UCAM-CL-TR-530, University of Cambridge, UK.

Shulte im Walde, S. 2002. "Evaluating verb subcategorization frames learned by a German statistical grammar against manual definitions in the Duden Dictionary". In *Proceedings of the 10th EURALEX International Congress*, 187-197.

Witten, Ian H. and Eibe Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. 2nd Edition, Morgan Kaufmann, San Francisco.

Conquest – an Open-Source Dialog System for Conferences

Dan Bohus, Sergio Grau Puerto, David Huggins-Daines, Venkatesh Keri, Gopala Krishna, Rohit Kumar, Antoine Raux, Stefanie Tomko
School of Computer Science
Carnegie Mellon University
{ dbohus, sgrau, dhuggins, vkeri, gopalakr, rohitk, antoine, stef }@ cs.cmu.edu

Abstract

We describe ConQuest, an open-source, reusable spoken dialog system that provides technical program information during conferences. The system uses a transparent, modular and open infrastructure, and aims to enable applied research in spoken language interfaces. The conference domain is a good platform for applied research since it permits periodical redeployments and evaluations with a real user-base. In this paper, we describe the system's functionality, overall architecture, and we discuss two initial deployments.

1 Introduction

Conducting applied spoken language interface research is generally a costly endeavor. Developing, deploying and maintaining real-world spoken language interfaces requires an existing infrastructure, a significant amount of engineering effort, and can greatly benefit from the availability of certain resources such as transcribed in-domain data.

In an effort to enable applied research and to lower this high cost of entry, we have developed ConQuest (**Con**ference **Quest**ions) an open-source spoken dialog system that provides access to schedule and technical program information during conferences. We believe the conference domain has a number of good properties for applied research: it includes a number of tasks of different complexities, it provides regular access to a real-world user population; it permits periodical redeployments and evaluations and therefore can provide a natural common evaluation task for the spoken language interfaces community.

The ConQuest system is constructed on top of the open, transparent and modular Olympus dialog system framework (2007), and can be easily reused across different conferences. To date, the system has been deployed in two conferences: InterSpeech 2006 and IJCAI 2007. Together with corpora collected from these deployments, the system is freely available for download (Conquest, 2007).

We begin by describing the ConQuest functionality in the next section. Then, in section 3 we provide an overview of the system architecture and discuss the development process. In section 4 we briefly discuss the two deployment efforts. Finally, in section 5 we discuss related work and draw a number of conclusions.

2 Functionality

As Figure 1 illustrates, ConQuest is a mixed-initiative spoken dialog system that provides access to schedule and technical program information during conferences.

Users can browse the schedule and find details about various papers or sessions of interest by providing identifying information, such as topics, session names, special events, paper titles, author names, specific dates and times, specific locations, or a combination thereof (e.g. turns 2, 4, 14). Additionally, the system also allows uses to listen to current announcements and to cast their votes for a Best Paper Award (e.g. turns 10-17).

The interaction is mixed-initiative; at any point, the user may switch the current focus of the conversation (e.g. turn 8). ConQuest employs an adaptive information presentation strategy that allows

Proceedings of NAACL HLT 2007, Companion Volume, pages 9–12,
Rochester, NY, April 2007.

1 S: Welcome to ConQuest […] I can give you information about papers, sessions, special events and announcements at Inter-Speech 2006. You can also cast a vote for the best paper award. How can I help you?
2 *U: are there any papers on spoken dialog systems today*
3 S: There are 23 papers on spoken dialog systems today. These papers are being presented in 3 sessions: Dialog and Discourse, Language Modeling for Spoken Dialog Systems and Spoken Dialog Systems I. To browse this list of papers, please tell me the author name, the paper title, the time or the venue you're looking for. If you're not interested in any of these papers say 'quit'.
4 *U: which ones are in the afternoon*
5 S: There will be 11 papers on spoken dialog systems this afternoon. Would you like to hear information about all these papers?
6 *U: yes*
7 S: The list of papers on spoken dialog systems this afternoon is: Paper 1 - "Dynamic Extension of a Grammar-based Dialogue System: Constructing an All-Recipes Knowing Robot", Paper 2 – […] Would you like more information about any of these?
8 *U: tell me more about the third one*
9 S: Paper 3 is titled "System- versus user-initiative dialog strategy for driver information systems", and was presented yesterday at 2 pm in the poster session in Allegheny Ballroom Foyer. This paper was written by Chantal Ackermann, and Marion Libossek. Would you like more information about any of these?
8 *U: no when is the banquet*
9 S: The Banquet will be held Wednesday from 7pm to 9:30 pm in the Heintz History Center. The Heinz History Center is located on Smallman street between 12th and 13th streets. The History Center is a museum that relates and interprets 250 years of history in the Western Pennsylvania area. Is there anything else I can do for you?
10 *U: yes I'd like to cast a vote*
11 S: Please tell me the title of the paper you'd like to vote for..
12 *U: I don't really know the title but I think it had Diane as one of the main authors* → [non-understanding]
13 S: Sorry, I didn't catch that. Please dial the first six letters from the title of the paper you would like to vote for.
14 *U: I don't know it it's a paper by Diane Litman*
15 S: There are 3 papers by Diane Litman. Paper 1 – "Discourse Structure and Speech Recognition Problems", Paper 2 – […], Paper 3 – […] Would you like to vote for any of these?
16 *U: Discourse Structure and Speech Recognition Problems*
17 S: Okay, I've recorded your vote for the paper entitled "Discourse Structure and Speech Recognition Problems" What else can I do for you?
18 *U: […]*

Figure 1. A sample interaction with ConQuest

users to easily navigate the schedule (see turns 3, 5 and 15). The system uses a rich repertoire of error recovery strategies to handle potential errors, including several fall-back strategies (e.g. turn 13).

3 System Architecture

The ConQuest system was built using RavenClaw/ Olympus (2007), an open-source framework that facilitates research and development in task oriented conversational spoken language interfaces. Olympus consists of a collection of components for recognition, language understanding, dialog management, language generation, speech synthesis, etc., and the corresponding communication infrastructure. To date, Olympus has been used to develop and deploy a number of other systems spanning different domains and interaction types (Bohus and Rudnicky, 2003).

A key characteristic of the Olympus framework is a clear separation between the domain independent programs (or components) and domain specific resources. This decoupling promotes reusability and significantly lessens the system development effort. In ConQuest, the authoring effort was focused on developing resources such as the lexicon, language model, grammar, dialog task specification, etc. Some interesting, unanticipated engineering challenges we faced during development were dealing with foreign names and accented characters and performing text normalization on various fields (e.g. Alex Smith and Alexander Smith are the same author), while at the same time ensuring consistency between these various resources. Below, we briefly comment of each component and the corresponding resources. Figure 2 provides a top-level architectural view.

Speech Recognition. ConQuest uses a recognition server coupled to a set of parallel recognition engines: two SPHINX-II decoders (Huang et al., 1992) that use gender-specific acoustic models, and a DTMF (touch-tone decoder). Each recognition engine uses class-based (e.g. paper titles, author names, etc.), state-specific trigram-language models. We started with an initial language model built using data collected with an early text-only prototype. We then internally deployed a speech based system, collected more data, transcribed it, and used it to retrain the language models. The

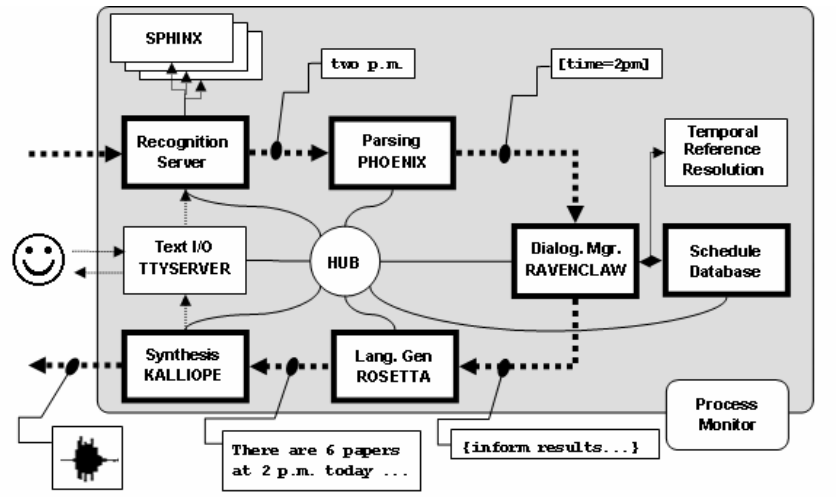


Figure 2. The Olympus dialog system reference architecture (a typical system)

final language models used during the InterSpeech deployment were trained from on a corpus of 6350 utterances. The system operated with a lexicon of 4795 words, which included 659 lexicalized (concatenated) paper titles, and 1492 lexicalized author names, and 78 lexicalized session names. The pronunciations were generated using CMU Dictionary and later manually corrected.

Language understanding. The system uses the Phoenix (Ward and Issar, 1994) robust parser to extract concepts from the recognition results. A domain-specific shallow semantic grammar was developed and concatenated with a domain-independent grammar for generic expressions like [Yes], [No], [Date], [Time], etc.

Dialog management. ConQuest uses a RavenClaw-based dialog manager (Bohus and Rudnicky, 2003). We developed a dialog task specification for the conference schedule domain, expressed as a hierarchical plan for the interaction, which the RavenClaw engine uses to drive the dialog. In the process, the RavenClaw engine automatically provides additional generic conversational skills such as error recovery strategies and support for various universal dialog mechanisms (e.g. repeat, start-over, what-can-I-say, etc.)

Backend/Database. A backend agent looks up schedule information from the database (stored as a flat text file). The backend agent also performs domain specific pre-lookup normalization (e.g. mapping author names to their canonical forms), and post-lookup processing of the returned records (e.g. clustering papers by sessions). The database file serves as starting point for constructing a number of other system resources (e.g. language model classes, lexicon, etc.)

Temporal reference resolution agent. Apart from the database agent, the dialog manager also communicates with an agent that resolves temporal expressions (e.g. tomorrow at four p.m.) into canonical forms.

Language generation. ConQuest uses Rosetta, a template-based language generation component. The authoring effort at this level consisted of writing various templates for the different system questions and information presentation prompts.

Speech synthesis. ConQuest uses the Cepstral (2005) speech synthesis engine, configured with an open-domain unit selection voice. We manually checked and corrected pronunciations for author names, various technical terms and abbreviations.

4 Development and Deployment

The first development of ConQuest system was done for the Interspeech 2006 conference held in Pittsburgh, PA. The iterative development process involved regular interaction with potential users i.e. researchers who regularly attend conferences. Seven developers working half time participated in this development for about three months. An estimated one man-year of effort was spent. This estimate does not include the effort involved in transcribing the data collected after the conference.

Two systems were deployed at the Interspeech 2006 conference: a desktop system using a close-talking microphone placed by the registration desk, and a telephone-based system. Throughout the conference we collected a corpus of 174 sessions. We have orthographically transcribed the user ut-

terances and are currently analyzing the data; we plan to soon release it to the community, together with detailed statistics, the full system logs as well as the full system source code (Conquest, 2007).

Following Interspeech 2006, ConQuest was redeployed at IJCAI 2007 conference held in Hyderabad, India. The second deployment took an estimated two man-months: three developers working half-time for over a month. The significant parts of the second deployment involved incorporating scheduling data for the IJCAI 2007 and implementing two new requirements i.e. support for workshops and Indian English speech recognition. The IJCAI development had fewer iterations than the first effort. The two desktop systems set up at the conference venue collected 129 sessions of data. This data is currently being transcribed and will soon be released to the community through the Conquest website (Conquest, 2007).

Through these two deployments of ConQuest the system specifications have been refined and we expect the development time to asymptote to less than a month after a few more deployments.

5 Discussion and Conclusion

Our primary goal in developing ConQuest was to enable research by constructing and releasing an open-source, full-fledged dialog system, as well as an initial corpus collected with this system. The system is built on top of an open, transparent and modular infrastructure that facilitates research in spoken language interfaces (Olympus, 2007).

There have been a number of other efforts to collect and publish dialog corpora, for instance within the DARPA Communicator project. A more recent project, that operates in a domain similar to ConQuest is DiSCoH, a Dialog System for Conference Help developed by researchers at AT&T, ICSI and Edinburgh University, and deployed during the SLT-2006 workshop (Adreani et al., 2006). While their goals are similar, i.e. to enable research, DiSCoH and ConQuest differ in a number of dimensions. Functionality-wise, DiSCoH offers general conference information about the venue, accommodation options and costs, paper submission, etc., while ConQuest provides access to the technical schedule and allows participants to vote for a best paper award. DiSCoH is built using AT&T technology and a call-routing approach; ConQuest relies on a plan-based dialog management framework (RavenClaw) and an open-source infrastructure (Olympus). Finally, the DiSCoH effort aims to develop a richly annotated dialog corpus to be used for research; ConQuest's aim is to provide both the full system and an initial transcribed and annotated corpus to the community.

The conference domain is interesting in that it allows for frequent redeployment and in theory provides regular access to a certain user-base. It should therefore facilitate research and periodical evaluations. Unfortunately, the dialog corpora collected so far using DiSCoH and ConQuest have been somewhat smaller than our initial expectations. We believe this is largely due to the fact that the systems provide information that is already accessible to users by other means (paper conference program, web-sites, etc.). Perhaps combining the functionalities of these two systems, and expanding into directions where the system provides otherwise hard-to-access information (e.g. local restaurants, transportation, etc.) would lead to increased traffic.

References

Adreani, G., Di Fabbrizio, G., Gilbert, M., Gillick, D., Hakkani-Tur, D., and Lemon, O., 2006 *Let's DiSCoH: Collecting an Annotated Open Corpus with Dialogue Acts and Reward Signals for Natural Language Helpdesk*, in Proceedings of IEEE SLT-2006 Workshop, Aruba Beach, Aruba.

Bohus, D., and Rudnicky, A., 2003. *RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda,* in Proceedings of Eurospeech 2003, Geneva, Switzerland.

Cepstral, LLC, 2005, Swift™: Small Footprint Text-to-Speech Synthesizer, http://www.cepstral.com.

Conquest, 2007, http://www.conquest-dialog.org.

Huang, X., Alleva, F., Hon, H.-W., Hwang, M.-Y., Lee, K.-F. and Rosenfeld, R., 1992. *The SPHINX-II Speech Recognition System: an overview*, in Computer Speech and Language, 7(2), pp 137-148, 1992.

Olympus/RavenClaw web page, as of January 2007: http://www.ravenclaw-olympus.org/.

Ward, W., and Issar, S., 1994. *Recent improvements in the CMU spoken language understanding system,* in Proceedings of the ARPA Human Language Technology Workshop, pages 213–216, Plainsboro, NJ.

JOINT VERSUS INDEPENDENT PHONOLOGICAL FEATURE MODELS WITHIN CRF PHONE RECOGNITION

Ilana Bromberg[*], **Jeremy Morris**[†], **and Eric Fosler-Lussier**[*†]
[*]Department of Linguistics
[†]Department of Computer Science and Engineering
The Ohio State University, Columbus, OH
bromberg@ling.ohio-state.edu, {morrijer, fosler}@cse.ohio-state.edu

Abstract

We compare the effect of joint modeling of phonological features to independent feature detectors in a Conditional Random Fields framework. Joint modeling of features is achieved by deriving phonological feature posteriors from the posterior probabilities of the phonemes. We find that joint modeling provides superior performance to the independent models on the TIMIT phone recognition task. We explore the effects of varying relationships between phonological features, and suggest that in an ASR system, phonological features should be handled as correlated, rather than independent.

1 Introduction

Phonological features have received attention as a linguistically-based representation for sub-word information in automatic speech recognition. These sub-phonetic features allow for a more refined representation of speech by allowing for temporal desynchronization between articulators, and help account for some phonological changes common in spontaneous speech, such as devoicing (Kirchhoff, 1999; Livescu, 2005). A number of methods have been developed for detecting acoustic phonological features and related acoustic landmarks directly from data using Multi-Layer Perceptrons (Kirchhoff, 1999), Support Vector Machines (Hasegawa-Johnson et al., 2005; Sharenborg et al., 2006), or Hidden Markov Models (Li and Lee, 2005). These techniques typically assume that acoustic phonological feature events are independent for ease of modeling.

In one study that broke the independence assumption (Chang et al., 2001), the investigators developed *conditional detectors*: MLP detectors of acoustic phonological features that are hierarchically dependent on a different phonological class. In (Rajamanohar and Fosler-Lussier, 2005) it was shown that such a conditional training of detectors tended to have correlated frame errors, and that improvements in detection could be obtained by training joint detectors. For many features, the best detector can be obtained by collapsing MLP phone posteriors into feature classes by marginalizing across phones within a class. This was shown only for frame-level classification rather than phone recognition.

Posterior estimates of phonological feature classes, as in Table 1, particularly those derived from MLPs, have been used as input to HMMs (Launay et al., 2002), Dynamic Bayesian Networks (DBNs) (Frankel et al., 2004; Livescu, 2005), and Conditional Random Fields (CRFs) (Morris and Fosler-Lussier, 2006). Here we evaluate phonological feature detectors created from MLP phone posterior estimators (joint feature models) rather than the independently trained MLP feature detectors used in previous work.

2 Conditional Random Fields

CRFs (Lafferty et al., 2001) are a joint model of a label sequence conditioned on a set of inputs. No independence is assumed among the input; the CRF model discriminates between hypothesized label sequences according to an exponential function of weighted feature functions:

$$P(\mathbf{y}|\mathbf{x}) \propto \exp \sum_{\mathrm{i}} \left(\mathrm{S}(\mathbf{x}, \mathbf{y}, \mathrm{i}) + \mathrm{T}(\mathbf{x}, \mathbf{y}, \mathrm{i})\right) \quad (1)$$

Proceedings of NAACL HLT 2007, Companion Volume, pages 13–16,
Rochester, NY, April 2007.

Class	Feature Values
SONORITY	Vowel, Obstruent, Sonorant, Syllabic, Silence
VOICE	Voiced, Unvoiced, N/A
MANNER	Fricative, Stop, Stop-Closure, Flap, Nasal, Approximant, Nasalflap, N/A
PLACE	Labial, Dental, Alveolar, Palatal, Velar, Glottal, Lateral, Rhotic, N/A
HEIGHT	High, Mid, Low, Lowhigh, Midhigh, N/A
FRONT	Front, Back, Central, Backfront, N/A
ROUND	Round, Nonround, Roundnonround, Nonroundround, N/A
TENSE	Tense, Lax N/A

Table 1: Phonetic feature classes and associated values

where $P(\mathbf{y}|\mathbf{x})$ is the probability of label sequence y given an input frame sequence x, i is the frame index, and S and T are a set of state feature functions and a set of transition feature functions, defined as:

$$S(x, y, i) = \sum_j \lambda_j s_j(y, x, i), \quad \text{and} \tag{2}$$

$$T(x, y, i) = \sum_k \mu_k t_k(y_{i-1}, y_i, x, i) \tag{3}$$

where λ and μ are weights determined by the learning algorithm. In NLP applications, the component feature functions s_j and t_k are typically realized as binary indicator functions indicating the presence or absence of a feature, but in ASR applications it is more typical to utilize real-valued functions, such as those derived from the sufficient statistics of Gaussians (e.g., (Gunawardana et al., 2005)).

We can use posterior estimates of phone classes or phonological feature classes from MLPs as feature functions (inputs) within the CRF model. A more detailed description of this CRF paradigm can be found in (Morris and Fosler-Lussier, 2006), which shows that the results of phone recognition using CRFs is comparable to that of HMMs or Tandem systems, with fewer constraints being imposed on the model. State feature functions in our system are defined such that

$$s_{\phi,f}(y_i, \mathbf{x}, i) = \begin{cases} NN_f(x_i), & \text{if} y_i = \phi \\ 0, & otherwise \end{cases} \tag{4}$$

where the MLP output for feature f at time i is $NN_f(x_i)$. This allows for an association between a phone ϕ and a feature f (even if f is traditionally not associated with ϕ).

In this study, we experiment with different methods of generating these feature functions. In various experiments, they are generated by training MLP phone detectors, by evaluating the feature information inherent in the MLP phone posteriors, and by training independent MLPs to detect the various features within the classes described. The use of CRFs allows us to explore the dependencies among feature classes, as well as the usefulness of phone posteriors versus feature classes as inputs.

3 Experimental Setup

We use the TIMIT speech corpus for all training and testing (Garofolo et al., 1993). The acoustic data is manually labeled at the phonetic level, and we propagate this phonetic label information to every frame of data. For the feature analyses, we employ a lookup table that defines each phone in terms of 8 feature classes, as shown in Table 1. We extract acoustic features in the form of 12th order PLP features plus delta coefficients. We then use these as inputs to several sets of neural networks using the ICSI QuickNet MLP neural network software (Johnson, 2004), with the 39 acoustic features as input, a varying number of phone or feature class posteriors as output, and 1000 hidden nodes.

4 Joint Phone Posteriors vs. Independent Feature Posteriors

The first experiment contrasts joint versus independent feature modeling within the CRF system. We compare a set of phonological feature probabilities derived from the phone posteriors (a joint model) with MLP phone posteriors and with independently trained MLP phonological feature posteriors.

The inputs to the first CRF are sets of 61 state feature functions from the phonemic MLP posteriors, each function is an estimate of the posterior proba-

Input Type.	Phn. Accuracy	Phn. Correct
Phones	67.27	68.77
Features	65.25	66.65
Phn. → Feat.	66.45	67.94

Table 2: Results for Exp. 1: Phone and feature posteriors as input to the CRF phone recognition

bility of one phone. The inputs to the second CRF model are sets of 44 functions corresponding to the phonological features listed in Table 1. The CRF models are trained to associate these feature functions with phoneme labels, incorporating the patterns of variation seen in the MLPs.

The results show that phone-based posteriors produce better phone recognition results than independently-trained phonological features. This could be due in part to the larger number of parameters in the system, but it could also be due to the joint modeling that occurs in the phone classifier.

In order to equalize the feature spaces, we use the output of the phoneme classifier to derive phonological feature posteriors. In each frame we sum the MLP phone posteriors of all phones that contain a given feature. For instance, in the first frame, for the feature LOW, we sum the posterior estimates attributed to the phones *aa*, *ae* and *ao*. This is repeated for each feature in each frame. The CRF model is trained on these data and tested accordingly. The results are significantly better ($p \leq .001$) than the previous features model, but are significantly worse than the phone posteriors ($p \leq .005$).

The results of Experiment 1 confirm the hypothesis of (Rajamanohar and Fosler-Lussier, 2005) that joint modeling using several types of feature information is superior to individual modeling in phone recognition, where only phoneme information is used. The difference between the phone posteriors and individual feature posteriors seems to be related both to the larger CRF parameter space with larger input, and the joint modeling provided by phone posteriors.

5 Phonological Feature Class Analysis

In the second experiment, we examine the influence of each feature class on the accuracy of the recognizer. We iteratively remove the set of state feature functions corresponding to each feature class from the input to the CRF. The original functions are the output of the independently-trained feature class MLPs. The phone recognition accuracy for the CRF having removed each class is shown in Table 3. In Table 4 we show how removing each feature class affects the labeling of vowels and consonants.

Class Removed	Feats.	Phn. Acc.	Phn. Corr.
None	44	65.25	66.65
Sonority	39	65.15	66.58
Voice	41	63.60*	65.03*
Manner	36	58.92*	60.60*
Place	35	53.22*	55.13*
Height	38	62.58*	64.07*
Front	39	64.51*	65.95*
Round	39	65.19	66.64
Tense	41	64.20*	65.65*
* $p \leq .05$, different from no features removed			

Table 3: Results of Exp. 2: Removing feature classes from the input

Manner provides an example of the influence of a single feature class. Both the Accuracy and Correctness scores decrease significantly when features associated with Manner are removed. Manner features distinguish consonants but not vowels, so the effect is concentrated on the recognition of consonants.

The results of Experiment 2 show that certain feature classes are redundant from the point of view of phone recognition. In English, Round is correlated with Front. When we remove Round, the phonemes remain uniquely identified by the other classes. The same is true for the Sonority class. The results show that the inclusion of these redundant features is not detrimental to the recognition accuracy. Accuracy and Correctness improve non-significantly when the redundant features are included.

Clearly, the "independent" phonological feature streams are not truly independent. Otherwise, performance would decrease overall as we removed each feature class, assuming predictiveness.

Removal of Place causes a slight worsening of recognition of vowels. This is surprising, because Place does not characterize vowels. An analysis of the MLP activations showed that the detector for Place=N/A is a stronger indicator for vowels than is the Sonority=Vowel detector. This is especially true for the vowel *ax*, which is frequent in the data,

Class Removed	Percent Correct:	
	Vowels	Consonants
None	62.68	68.91
Sonority	62.18	69.08
Voice	62.39	66.53*
Manner	61.84	59.89*
Place	60.77*	51.94*
Height	55.92*	68.69
Frontness	60.80*	68.87
Roundness	62.25	69.13
Tenseness	60.15*	68.76
* $p \leq .05$, different from no features removed		

Table 4: Effect of removing each feature class on recognition accuracy of vowels and consonants

thus greatly influences the vowel recognition statistic. Removing the Place detectors leads to a loss in vowel vs. consonant information. This results in an increased number of consonant for vowel substitutions (from 560 to 976), thus a decrease in vowel recognition accuracy.

Besides extending the findings in (Rajamanohar and Fosler-Lussier, 2005), this provides a cautionary tale for incorporating redundant phonological feature estimators into ASR: these systems need to be able to handle correlated input, either by design (as in a CRF), through full or semi-tied covariance matrices in HMMs, or by including the appropriate statistical dependencies in DBNs.

6 Summary

We have shown the effect of using joint modeling of phonetic feature information in conjunction with the use of CRFs as a discriminative classifier. Phonetic posteriors, as joint models of phonological features, provide superior phone recognition performance over independently-trained phonological feature models. We also find that redundant features are often modeled well within the CRF framework.

7 Acknowledgments

The authors thank the International Computer Science Institute for providing the neural network software. The authors also thank four anonymous reviewers. This work was supported by NSF ITR grant IIS-0427413; the opinions and conclusions expressed in this work are those of the authors and not of any funding agency.

References

S. Chang, S. Greenberg, and M. Wester. 2001. An elitist approach to articulatory-acoustic feature classification. In *Interspeech*.

J. Frankel, M. Wester, and S. King. 2004. Articulatory feature recognition using dynamic bayesian networks. In *ICSLP*.

J. Garofolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallett, and N. Dahlgren. 1993. DARPA TIMIT acoustic-phonetic continuous speech corpus. Technical Report NISTIR 4930, National Institute of Standards and Technology, Gaithersburg, MD, February. Speech Data published on CD-ROM: NIST Speech Disc 1-1.1, October 1990.

A. Gunawardana, M. Mahajan, A. Acero, and J. Platt. 2005. Hidden conditional random fields for phone classification. In *Interspeech*.

M. Hasegawa-Johnson et al. 2005. Landmark-based speech recognition: Report of the 2004 Johns Hopkins Summer Workshop. In *ICASSP*.

D. Johnson. 2004. ICSI Quicknet software package. http://www.icsi.berkeley.edu/Speech/qn.html.

K. Kirchhoff. 1999. *Robust Speech Recognition Using Articulatory Information*. Ph.D. thesis, University of Bielefeld.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*.

B. Launay, O. Siohan, A. C. Surendran, and C.-H. Lee. 2002. Towards knowledge based features for large vocabulary automatic speech recognition. In *ICASSP*.

J. Li and C.-H. Lee. 2005. On designing and evaluating speech event detectors. In *Interspeech*.

K. Livescu. 2005. *Feature-Based Pronunciation Modeling for Automatic Speech Recognition*. Ph.D. thesis, MIT.

J. Morris and E. Fosler-Lussier. 2006. Combining phonetic attributes using conditional random fields. In *Interspeech*.

M. Rajamanohar and E. Fosler-Lussier. 2005. An evaluation of hierarchical articulatory feature detectors. In *IEEE Automatic Speech Recogntion and Understanding Workshop*.

O. Sharenborg, V. Wan, and R.K. Moore. 2006. Capturing fine-phonetic variation in speech through automatic classification of articulatory features. In *ITRW on Speech Recognition and Intrinsic Variation*.

K-Best Suffix Arrays

Kenneth Church
Microsoft
One Microsoft Way
Redmond, WA 98052
church@microsoft.com

Bo Thiesson
Microsoft
One Microsoft Way
Redmond, WA 98052
thiesson@microsoft.com

Robert Ragno
Microsoft
One Microsoft Way
Redmond, WA 98052
rragno@microsoft.com

Abstract

Suppose we have a large dictionary of strings. Each entry starts with a figure of merit (popularity). We wish to find the *k*-best matches for a substring, *s*, in a dictionary, *dict*. That is, grep *s dict* | sort –n | head –*k*, but we would like to do this in sublinear time. Example applications: (1) web queries with popularities, (2) products with prices and (3) ads with click through rates. This paper proposes a novel index, *k-best suffix arrays*, based on ideas borrowed from suffix arrays and kd-trees. A standard suffix array sorts the suffixes by a single order (lexicographic) whereas k-best suffix arrays are sorted by two orders (lexicographic and popularity). Lookup time is between *log N* and *sqrt N*.

1 Standard Suffix Arrays

This paper will introduce *k-best suffix arrays*, which are similar to standard suffix arrays (Manber and Myers, 1990), an index that makes it convenient to compute the frequency and location of a substring, *s*, in a long sequence, *corpus*. A suffix array, *suf*, is an array of all *N* suffixes, sorted alphabetically. A suffix, *suf*[*i*], also known as a semi-infinite string, is a string that starts at position *j* in the corpus and continues to the end of the corpus. In practical implementations, a suffix is a 4-byte integer, *j*. In this way, an int (constant space) denotes a long string (*N* bytes).

The *make_standard_suf* program below creates a standard suffix array. The program starts with a *corpus*, a global variable containing a long string of *N* characters. The program allocates the suffix array *suf* and initializes it to a vector of *N* ints (suffixes) ranging from 0 to *N*−1. The suffix array is sorted by lexicographic order and returned.

```
int* make_standard_suf () {
  int N = strlen(corpus);
  int* suf = (int*)malloc(N * sizeof(int));
  for (int i=0; i<N; i++) suf[i] = i;
  qsort(suf, N, sizeof(int), lexcomp);
  return suf;}

int lexcomp(int* a, int* b)
{ return strcmp(corpus + *a, corpus + *b);}
```

This program is simple to describe (but inefficient, at least in theory) because *strcmp* can take O(*N*) time in the worst case (where the corpus contains two copies of an arbitrarily long string). See http://cm.bell-labs.com/cm/cs/who/doug/ssort.c for an implementation of the O(*N* log *N*) Manber and Myers algorithm. However, in practice, when the corpus is a dictionary of relatively short entries (such as web queries), the worst case is unlikely to come up. In which case, the simple *make_suf* program above is good enough, and maybe even better than the O(*N* log *N*) solution.

1.1 Standard Suffix Array Lookup

To compute the frequency and locations of a substring *s*, use a pair of binary searches to find *i* and *j*, the locations of the first and last suffix in the suffix array that start with *s*. Each suffix between *i* and *j* point to a location of *s* in the corpus. The frequency is simply: $j - i + 1$.

Here is some simple code. We show how to find the first suffix. The last suffix is left as an exercise. As above, we ignore the unlikely worst

Proceedings of NAACL HLT 2007, Companion Volume, pages 17–20,
Rochester, NY, April 2007.

case (two copies of a long string). See references mentioned above for worst case solutions.

```
void standard_lookup(char* s, int* suf, int N){
  int* i = find_first_suf(s, suf, N);
  int* j = find_last_suf(s, suf, N);
  for (int* k=i; k<=j; k++) output(*k);}

int* find_first_suf(char* s, int* suf, int N) {
  int len = strlen(s);
  int* high = suf + N;
  while (suf + 2 < high) {
    int* mid = suf + (high−suf)/2;
    int c = strncmp(s, corpus + *mid, len);
    if (c == 0) high = mid+1;
    else if (c < 0) high = mid;
    else suf = mid;}
  for ( ; suf < high; suf++)
    if (strncmp(s, corpus + *suf, len) == 0)
      return suf;
  return NULL;}          // not found
```

2 K-Best Suffix Arrays

K-best suffix arrays are like standard suffix arrays, except there are two orders instead of one. In addition to lexicographic order, we assume a figure of merit, which we will refer to as popularity. For example, the popularity of a string could be its frequency in a search log. The code below assumes that the corpus is a sequence of strings that comes pre-sorted by popularity, and then the popularities have been stripped off. These assumptions make it very easy to compare two strings by popularity. All *popcomp* has to do is to compare the two positions in the corpus.[1]

The *make_kbest_suf* program below is similar to the *make_standard_suf* program above except we now sort by the two orders at alternating depths in the tree. First we sort lexicographically and then we sort by popularity and so on, using a construction similar to KD-Trees (Bentley, 1975). The code below is simple to describe (though there are more efficient implementations that avoid unnecessary qsorts).

```
int* make_kbest_suf () {
  int N = strlen(corpus);
  int* suf = (int*)malloc(N * sizeof(int));
  for (int i=0; i<N; i++) suf[i]=i;
  process(suf, suf+N, 0);
  return suf;}

void process(int* start, int* end, int depth) {
  int* mid = start + (end − start)/2;
  if (end <= start+1) return;
  qsort(start, end-start, sizeof(int),
        (depth & 1) ? popcomp : lexcomp);
  process(start, mid, depth+1);
  process(mid+1, end, depth+1);}

int popcomp(int* a, int* b) {
  if (*a > *b) return 1;
  if (*a < *b) return −1;
  return 0;}
```

2.1 K-Best Suffix Array Lookup

To find the k-best matches for a particular substring *s*, we do what we would normally do for standard suffix arrays on lexicographic splits. However, on popularity splits, we search the more popular half first and then we search the less popular half, if necessary.

An implementation of *kbest-lookup* is given below. *D* denotes the depth of the search thus far. *Kbest-lookup* is initially called with *D* of 0. *Propose* maintains a heap of the k-best matches found thus far. *Done* returns true if its argument is less popular than the kth best match found thus far.

```
void kbest_lookup(char* s, int* suf, int N, int D){
  int* mid = suf + N/2;
  int len = strlen(s);

  if (N==1 && strncmp(s, corpus+*suf, len)==0)
      propose(*suf);
  if (N <= 1) return;

  if (D&1) {              // popularity split
    kbest_lookup(s, suf, mid−suf, D+1);
    if (done(*mid)) return;
    if (strncmp(s, corpus + *mid, len) == 0)
      propose(*mid);
    kbest_lookup(s, mid+1, (suf+N)−mid−1,
                           D+1);}
  else {                  // lexicographic split
    int c = strncmp(s, corpus + *mid, len);
    int n = (suf+N)−mid−1;
    if (c < 0) kbest_lookup(s, suf, mid-suf, D+1);
    else if (c > 0) kbest_lookup(s, mid+1, n, D+1);
    else { kbest_lookup(s, suf, mid-suf, depth+1);
           propose(*mid);
           kbest_lookup(s, mid+1, n, D+1); }}}
```

[1] With a little extra book keeping, one can keep a table on the side that makes it possible to map back and forth between popularity rank and the actual popularity. This turns out to be useful for some applications.

2.2 A Short Example: To be or not to be

Suppose we were given the text, "to be or not to be." We could then generate the following dictionary with frequencies (popularities).

Popularity	Word
2	to
2	be
1	or
1	not

The dictionary is sorted by popularity. We treat the second column as an N=13 byte corpus (with underscores at record boundaries): to_be_or_not_

Standard		K-Best	
suf	corpus + suf[i]	suf	corpus + suf[i]
12	_	2	_be_or_not_
2	_be_or_not_	3	be_or_not_
8	_not_	4	e_or_not_
5	_or_not_	5	_or_not_
3	be_or_not_	8	_not_
4	e_or_not_	12	_
9	not_	9	not_
1	o_be_or_not_	1	o_be_or_not_
6	or_not_	6	or_not_
10	ot_	0	to_be_or_not_
7	r_not_	7	r_not_
11	t_	10	ot_
0	to_be_or_not_	11	t_

The standard suffix array is the 1st column of the table above. For illustrative convenience, we show the corresponding strings in the 2nd column. Note that the 2nd column is sorted lexicographically.

The k-best suffix array is the 3rd column with the corresponding strings in the 4th column. The first split is a lexicographic split at 9 ("not_"). On both sides of that split we have a popularity split at 5 ("_or_not_") and 7 ("r_not_"). (Recall that relative popularity depends on corpus position.) Following there are 4 lexicographic splits, and so on.

If k-best lookup were given the query string *s* = "o," then it would find 1 (o_be_or_not_), 6 (or_not_) and 10 (ot_) as the best choices (in that order). The first split is a lexicographic split. All the matches are below 9 (not_). The next split is on popularity. The matches above this split (1&6) are as popular as the matches below this split (10).

It is often desirable to output matching records (rather than suffixes). Records are output in popularity order. The actual popularity can be output, using the side table mentioned in footnote 1:

Popularity	Record
2	to
1	or
1	not

2.3 Time and Space Complexity

The space requirements are the same for both standard and k-best suffix arrays. Both indexes are permutations of the same suffixes.

The time requirements are quite different. Standard suffix arrays were designed to find all matches, not the k-best. Standard suffix arrays can find all matches in O(*log N*) time. However, if we attempt to use standard suffix arrays to find the k-best, something they were not designed to do, then it could take a long time to sort through the worst case (an embarrassment of riches with lots of matches). When the query matches every string in the dictionary, standard suffix arrays do not help us find the best matches. K-best suffix arrays were designed to handle an embarrassment of riches, which is quite common, especially when the substring *s* is short. Each popularity split cuts the search space in half when there are lots of lexicographic matches.

The best case for k-best suffix arrays is when the popularity splits always work in our favor and we never have to search the less popular half. The worst case is when the popularity splits always fail, such as when the query string *s* is not in the corpus. In this case, we must always check both the popular half and the unpopular half at each split, since the failure to find a lexicographic match in the first tells us nothing about the existence of matches in the second.

Asymptotically, k-best lookup takes between log N and sqrt N time. To see this complexity result, let P(N) be the work to process N items starting with a popularity splits and let L(N) be the work to process N items starting with a lexicographic splits.

Thus,

$P(N) = \alpha L(N/2) + C_1$
$L(N) = P(N/2) + C_2$

where $\alpha = 2-p$, when p is the probability that the popular half contains sufficient matches. α lies between 1 (best case) and 2 (worst case). C_1 and C_2 are constants. Thus,

$$P(N) = \alpha\, P(N/4) + C \quad (1)$$

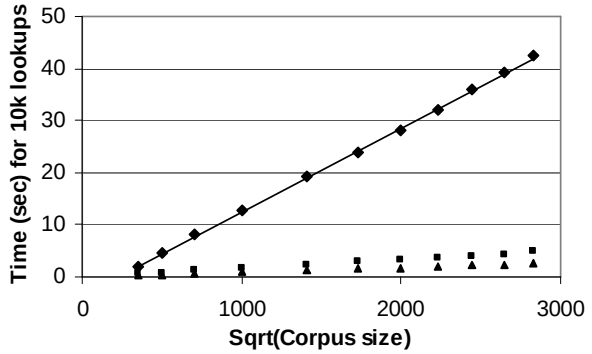
where $C = C_1 + \alpha C_2$. Using the master method (Cormen *et al*, 2001), $P(N) = O(log_2 N)$ in the best case (α=1). In the worst case (α=2), $P(N) = O(sqrt\ N)$. In general, for $\alpha > 1$, $P(N) = O(N^{(log_2 \alpha)/2})$.

In practical applications, we expect popularity splits to work more often than not, and therefore we expect the typical case to be closer to the best case than the worst case.

3 Empirical Study

The plot below shows the k-best lookup time as a function of square root of corpus size. We extracted sub-corpora from a 150 MB collection of 8M queries, sorted by popularity, according to the logs from Microsoft www.live.com. All experiments were performed on a Pentium 4, 3.2GHz dual processor machine with enough memory to avoid paging.

The line of diamonds shows the worst case, where we the query string is not in the index. Note that the diamonds fit the regression line quite well, confirming the theory in the previous section: The worst case lookup is *O*(*sqrt N*).



To simulate a more typical scenario, we constructed random samples of queries by popularity, represented by squares in the figure. Note that the squares are well below the line, demonstrating that these queries are considerably easier than the worst case.

K-best suffix arrays have been used in auto-complete applications (Church and Thiesson, 2005). The triangles with the fastest lookup times demonstrate the effectiveness of the index for this application. We started with the random sample above, but replaced each query q in the sample with a substring of q (of random size).

4 Conclusion

A new data structure, *k-best suffix arrays*, was proposed. K-best suffix arrays are sorted by two orders, lexicographic and popularity, which make it convenient to find the most popular matches, especially when there are lots of matches. In many applications, such as the web, there are often embarrassments of riches (lots of matches).

Lookup time varies from *log N* to *sqrt N*, depending on the effectiveness of the popularity splits. In the best case (e.g., very short query strings that match nearly everything), the popularity splits work nearly every time and we rarely have to search the less popular side of a popularity split. In this case, the time is close to *log N*. On the other hand, in the worst case (e.g., query strings that match nothing), the popularity splits never work, and we always have to search both sides of a popularity split. In this case, lookup time is *sqrt N*. In many cases, popularity splits work more often than not, and therefore, performance is closer to *log N* than *sqrt N*.

References

Jon Louis Bentley. 1975. Multidimensional Binary Search Trees Used for Associative Searching, *Communications of the ACM*, 18:9, pp. 509-517.

Kenneth Church and Bo Thiesson. 2005. The Wild Thing, *ACL*, pp. 93-96.

Udi Manber and Gene Myers. 1990. Suffix Arrays: A New Method for On-line String Searches, *SODA*, pp. 319-327.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2001. Introduction to Algorithms, Second Edition. *MIT Press and McGraw-Hill*, pp.73–90.

Translation Model Pruning via Usage Statistics for Statistical Machine Translation

Matthias Eck, Stephan Vogel, and Alex Waibel
InterACT Research
Carnegie Mellon University, Pittsburgh, USA
matteck@cs.cmu.edu, vogel+@cs.cmu.edu, ahw@cs.cmu.edu

Abstract

We describe a new pruning approach to remove phrase pairs from translation models of statistical machine translation systems. The approach applies the original translation system to a large amount of text and calculates usage statistics for the phrase pairs. Using these statistics the relevance of each phrase pair can be estimated. The approach is tested against a strong baseline based on previous work and shows significant improvements.

1 Introduction

A relatively new device for translation systems are small portable devices like cell phones, PDAs and handheld game consoles. The idea here is to have a lightweight and convenient translation device e.g. for tourists that can be easily carried. Other applications include medical, relief, and military scenarios.

Preferably such a device will offer speech-to-speech translation for both (or multiple) translation directions. These devices have been researched and are starting to become commercially available (e.g. Isotani et al., 2003). The main challenges here are the severe restrictions regarding both memory and computing power on such a small portable device.

1.1 Statistical Machine Translation

Generally statistical machine translation systems have recently outperformed other translation approaches so it seems natural to also apply them in these scenarios.

A main component of every statistical machine translation system is the translation model. The translation model assigns translation probabilities to phrase[1] pairs of source and target phrases extracted from a parallel bilingual text. These phrase pairs are applied during the decoding process and their target sides are combined to form the final translation. A variety of algorithms to extract phrase pairs has been proposed. (e.g. Och and Ney, 2000 and Vogel, 2005).

Our proposed approach now tries to remove phrase pairs, which have little influence on the final translation performance, from a translation system (*pruning* of the translation model[2]). The goal is to reduce the number of phrase pairs and in turn the memory requirement of the whole translation system, while not impacting the translation performance too heavily.

The approach does not depend on the actual algorithm used to extract the phrase pairs and can be applied to every imaginable method that assigns probabilities to phrase pairs. We assume that the phrase pairs were pre-extracted before decoding. (in contrast to the proposed approaches to "online phrase extraction" (Zhang and Vogel, 2005; Callison-Burch et al., 2005)).

The task now is to remove enough pre-extracted phrase pairs in order to accommodate the possibly strict memory limitations of a portable device while restricting performance degradation as much as possible.

We will not specifically address the computing power limitations of the portable devices in this paper.

[1] A "phrase" here can also refer to a single word.
[2] Small language models are also desirable and the approaches could be applied as well but this was not investigated yet.

Proceedings of NAACL HLT 2007, Companion Volume, pages 21–24,
Rochester, NY, April 2007.

2 Previous work

Previous work mainly introduced two natural ideas to prune phrase pairs. Both are for example directly available in the Pharaoh decoder (Koehn, 2004).

Probability threshold
A very simple way to prune phrase pairs from a translation model is to use a probability threshold and remove all pairs for which the translation probability is below the threshold. The reasoning for this is that it is very unlikely that a translation with a very low probability will be chosen (over another translation candidate with a higher probability).

Translation variety threshold
Another way to prune phrase pairs is to impose a limit on the number of translation candidates for a certain phrase. That means the pruned translation model can only have equal or fewer possible translations for a given source phrase than the threshold. This is accomplished by sorting the phrase pairs for each source phrase according to their probability and eliminating low probability ones until the threshold is reached.

3 Pruning via Usage Statistics

The approach presented here uses a different idea inspired by the *Optimal Brain Damage* algorithm for neural networks (Le Cun et al., 1990).

The Optimal Brain Damage algorithm for neural networks computes a *saliency* for each network element. The saliency is the relevance for the performance of the network. In each pruning step the element with the smallest saliency is removed, and the network is re-trained and all saliencies are recalculated etc.

We can analogously view each phrase pair in the translation system as such a network element. The question is of course how to calculate the relevance for the performance for each phrase pair.

A simple approximation was already done in the previous work using a probability or variety threshold. Here the relevance is estimated using the phrase pair probability or the phrase pair rank as relevance indicators.

But these are not the only factors that influence the final selection of a phrase pair and most of these factors are not established during the training and phrase extraction process. Especially the following two additional factors play a major role in the importance of a phrase pair.

Frequency of the source phrase
We can clearly say that a phrase pair with a very common source phrase will be much more important than a phrase pair where the source phrase occurs only very rarely.

Actual use of the phrase-pair
But even phrase-pairs with very common source phrases might not be used for the final translation hypothesis. It is for example possible that it is part of a longer phrase pair that gets a higher probability so that the shorter phrase pair is not used.

Generally there are a lot of different factors influencing the estimated importance of a phrase pair and it seems hard to consider every influence separately. Hence the proposed idea does not use a combination of features to estimate the phrase pair importance. Instead the idea is to just apply the translation system to a large amount of text and see how often a phrase pair is actually used (i.e. influences the translation performance). If the translated text is large enough this will give a good statistics of the relevance of this respective phrase pair. This leads to the following algorithm:

Algorithm
Translate a large amount of (in-domain) data with the translation system (tuned on a development set) and collect the following two statistics for each phrase pair in the translation model.

- *c(*phrase pair*)* = Count how often a phrase pair was *considered* during decoding (i.e. was added to the translation lattice)
- *u(*phrase pair*)* = Count how often a phrase pair was *used* in the final translation (i.e. in the chosen path through the lattice).

The overall score for a phrase pair with simple smoothing (+1) is calculated as:

$$score(\text{phrase pair}) = \left[\log(c(\text{phrase pair})+1)\right] * \left[u(\text{phrase pair})+1\right]$$

We use the logarithm function to limit the influence of the *c* value. The *u* value is more important as this measures how often a phrase was actually used in a translation hypothesis. This scoring func-

tion was empirically found after experimenting with a variety of possible scoring terms.

The phrase pairs can then be sorted according to this score and the top *n* phrase pairs can be selected for a smaller phrase translation model.

4 Data and Experiments

4.1 Experimental Setup & Baseline

Translation system
The translation system that was used for the experiments is a state-of-the-art statistical machine translation system (Eck et al. 2006). The system uses a phrase extraction method described in Vogel (2005) and a 6-gram language model.

Training and testing data
The training data for all experiments consisted of the BTEC corpus (Takezawa et al., 2002) with 162,318 lines of parallel Japanese-English text. All translations were done from Japanese to English. The language model was trained on the English part of the training data.

The test set from the evaluation campaign of IWSLT 2004 (Akiba et al., 2004) was used as testing data. This data consists of 500 lines of tourism data. 16 reference translations to English were available.

Extracted phrases
Phrase pairs for n-grams up to length 10 were extracted (with low frequency thresholds for higher n-grams). This gave 4,684,044 phrase pairs (273,459 distinct source phrases). The baseline score using all phrase pairs was 59.11 (BLEU, Papineni et al., 2002) with a 95% confidence interval of [57.13, 61.09].

Baseline pruning
The approaches presented in previous work served as a baseline. The probability threshold was tested for 8 values (0 (no pruning), 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1) while the variety threshold tested for 14 values (1, 2, 3, 4, 5, 6, 8, 10, 15, 20, 50, 100, 200, 500 (no pruning in this case)) and all combinations thereof. The final translation scores for different settings are very fluctuating. For that reason we defined the baseline score for each possible size as the best score that was reached with equal or less phrase pairs than the given size in any of the tested combinations.

4.2 Results for Pruning via Usage Statistics

For the proposed approach "Pruning via Usage Statistics", the translation system was applied to the 162,318 lines of Japanese training data.

As explained in section 3 it was now counted for each phrase pair how often it occurred in a translation lattice and how often it was used for the final translation. The phrase pairs were then sorted according to their relevance estimation and the top *n* phrase pairs were chosen for different values of *n*. The pruned phrase table was then used to translate the IWSLT 2004 test set. Table 1 shows the results comparing the baseline scores with the results using the described pruning. Figure 1 illustrates the scores. The plateaus in the baseline graph are due to the baseline definition as stated above.

	BLEU scores		
# of Phrase Pairs (*n*)	Baseline	Pruning	Relative score improvement
100,000	-	0.4735	-
200,000	0.3162	0.5008	58.38%
300,000	0.4235	0.5154	21.70%
400,000	0.4743	0.5241	10.50%
500,000	0.4743	0.5269	11.09%
600,000	0.4890	0.5359	9.59%
800,000	0.5194	0.5394	3.85%
1,000,000	0.5355	0.5442	1.62%
1,500,000	0.5413	0.5523	2.03%
2,000,000	0.5630	0.5749	2.11%
3,000,000	0.5778	0.5798	0.35%
4,000,000	0.5855	0.5865	0.17%
4,684,044	0.5911	0.5911	0.00%

Table 1: BLEU scores at different levels of pruning (Baseline: Best score with equal or less phrase pairs)

For more than 1 million phrase pairs the differences are not very pronounced. However the translation score for the proposed pruning algorithm is still not significantly lower than the 59.11 score at 2 million phrase pairs while the baseline drops slightly faster. For less than 1 million phrase pairs the differences become much more pronounced with relative improvements of up to 58% at 200,000 phrase pairs. It is interesting to note that the improved pruning removes infrequent source

phrases and to a lesser extent source vocabulary even for larger numbers of phrase pairs.

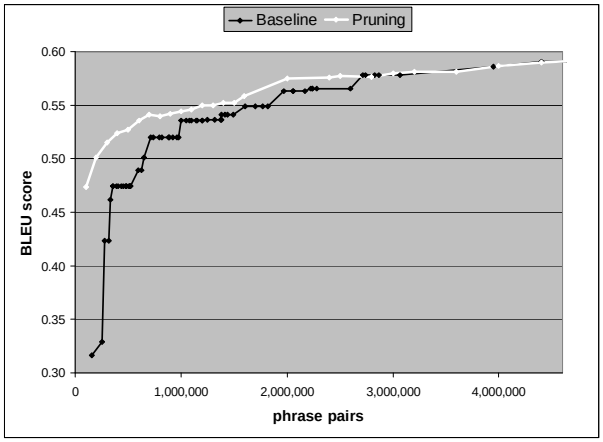


Figure 1: Pruning and baseline comparison

5 Conclusions and Future Work

The proposed pruning algorithm is able to outperform a strong baseline based on previously introduced threshold pruning ideas. Over 50% of phrase pairs can be pruned without a significant loss of performance. Even for very low memory situations the improved pruning remains a viable option while the baseline pruning performance drops heavily.

One idea to improve this new pruning approach is to exchange the *used* count with the count of the phrase occurring in the best path of the lattice according to a scoring metric. This would require having a reference translation available to be able to tell which path is the actual best one (metric-best path). It would be interesting to compare the performance if the statistics is done using the metric-best path on a smaller amount of data to the performance if the statistics is done using the model-best path on a larger amount (as there is no reference translation necessary).

The Optimal Brain Damage algorithm recalculates the *saliency* after removing each network element. It could also be beneficial to sequentially prune the phrase pairs and always re-calculate the statistics after removing a certain number of phrase pairs.

6 Acknowledgements

This work was partly supported by the US DARPA under the programs GALE and TRANSTAC.

7 References

Yasuhiro Akiba, Marcello Federico, Noriko Kando, Hiromi Nakaiwa, Michael Paul, and Jun'ichi Tsujii}. 2004. *Overview of the IWSLT04 Evaluation Campaign*. Proceedings of IWSLT 2004, Kyoto, Japan.

Chris Callison-Burch, Colin Bannard, and Josh Schroeder. 2005. *Scaling Phrase-Based Statistical Machine Translation to Larger Corpora and Longer Phrases.* Proceedings of ACL 2005, Ann Arbor, MI, USA.

Yann Le Cun, John S. Denker, and Sara A. Solla. 1990. *Optimal brain damage*. In Advances in Neural Information Processing Systems 2, pages 598-605. Morgan Kaufmann, 1990.

Matthias Eck, Ian Lane, Nguyen Bach, Sanjika Hewavitharana, Muntsin Kolss, Bing Zhao, Almut Silja Hildebrand, Stephan Vogel, and Alex Waibel. 2006. *The UKA/CMU Statistical Machine Translation System for IWSLT 2006.* Proceedings of IWSLT 2006, Kyoto, Japan.

Ryosuke Isotani, Kyoshi Yamabana, Shinichi Ando, Ken Hanazawa, Shin-ya Ishikawa and Ken.ichi Iso. 2003. *Speech-to-speech translation software on PDAs for travel conversation*. NEC research & development, Tokyo, Japan.

Philipp Koehn. 2004. *A Beam Search Decoder for Statistical Machine Translation Models.* Proceedings of AMTA 2004, Baltimore, MD, USA.

Franz Josef Och and Hermann Ney, 2000. *Improved statistical alignment models*, Proceedings of ACL 2000, Hongkong, China.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. *BLEU: a Method for Automatic Evaluation of Machine Translation.* Proceedings of ACL 2002, Philadelphia, PA, USA.

Toshiyuki Takezawa, Eiichiro Sumita, Fumiaki Sugaya, Hirofumi Yamamoto, and Seiichi Yamamoto. 2002. *Toward a Broad-coverage Bilingual Corpus for Speech Translation of Travel Conversation in the Real World.* Proceedings of LREC 2002, Las Palmas, Spain.

Stephan Vogel. 2005. *PESA: Phrase Pair Extraction as Sentence Splitting.* Proceedings of MTSummit X, Phuket, Thailand.

Ying Zhang and Stephan Vogel. 2005. *An Efficient Phrase-to-Phrase Alignment Model for Arbitrarily Long Phrases and Large Corpora*. Proceedings of EAMT 2005, Budapest, Hungary.

Combination of Statistical Word Alignments Based on Multiple Preprocessing Schemes

Jakob Elming
Center for Comp. Modeling of Language
Copenhagen Business School
je.id@cbs.dk

Nizar Habash
Center for Comp. Learning Systems
Columbia University
habash@cs.columbia.edu

Abstract

We present an approach to using multiple preprocessing schemes to improve statistical word alignments. We show a relative reduction of alignment error rate of about 38%.

1 Introduction

Word alignments over parallel corpora have become an essential supporting technology to a variety of natural language processing (NLP) applications most prominent among which is statistical machine translation (SMT).[1] Although phrase-based approaches to SMT tend to be robust to word-alignment errors (Lopez and Resnik, 2006), improving word-alignment is still useful for other NLP research that is more sensitive to alignment quality, e.g., projection of information across parallel corpora (Yarowsky et al., 2001).

In this paper, we present a novel approach to using and combining multiple preprocessing (tokenization) schemes to improve word alignment. The intuition here is similar to the combination of different preprocessing schemes for a morphologically rich language as part of SMT (Sadat and Habash, 2006) except that the focus is on improving the alignment quality. The language pair we work with is Arabic-English.

In the following two sections, we present related work and Arabic preprocessing schemes. Section 4 and 5 present our approach to alignment preprocessing and combination, respectively. Results are presented in Section 6.

[1]The second author was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023. Any opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of DARPA. We thank Necip Ayan, Mona Diab, Bonnie Dorr, Abe Ittycheriah, Martin Jansche and Owen Rambow for helpful discussions.

2 Related Work

Recently, several successful attempts have been made at using supervised machine learning for word alignment (Liu et al., 2005; Taskar et al., 2005; Ittycheriah and Roukos, 2005; Fraser and Marcu, 2006). In contrast to generative models, this framework is easier to extend with new features. With the exception of Fraser and Marcu (2006), these previous publications do not entirely discard the generative models in that they integrate IBM model predictions as features. We extend on this approach by including alignment information based on multiple preprocessing schemes in the alignment process.

In other related work, Tillmann et al. (1997) use several preprocessing strategies on both source and target language to make them more alike with regards to sentence length and word order. Lee (2004) only changes the word segmentation of the morphologically complex language (Arabic) to induce morphological and syntactic symmetry between the parallel sentences. We differ from these two in that we do not decide on a certain scheme to make source and target sentences more symmetrical. Instead, it is left to the alignment algorithm to decide under which circumstances alignment information based on a specific scheme is more likely to be correct than information based on other schemes.

3 Arabic Preprocessing Schemes

Arabic is a morphologically complex language with a large set of morphological features. As such, the set of possible preprocessing schemes is rather large (Habash and Sadat, 2006). We focus here on a subset of schemes pertaining to Arabic attachable clitics. There are three degrees of cliticization that apply to a word BASE: ([CONJ+ [PART+ [Al+ BASE +PRON]]]). At the deepest level, the BASE can have a definite article +ال (Al+ *the*)[2] or a member of the

[2]Arabic is transliterated in Buckwalter's transliteration scheme.

Proceedings of NAACL HLT 2007, Companion Volume, pages 25–28,
Rochester, NY, April 2007.

Table 1: Arabic preprocessing scheme variants for وسيكتبها 'and he will write it'

Preprocessing Scheme		Example	
AR	simple	وسيكتبها	wsyktbhA
$D1$	split CONJ	و+ سيكتبها	w+ syktbhA
$D2$	split CONJ, PART	و+ س+ يكتبها	w+ s+ yktbhA
TB	Arabic Treebank	و+ سيكتب +ها	w+ syktb +hA
$D3$	split all clitics	و+ س+ يكتب +ها	w+ s+ yktb +hA

class of pronominal clitics, +PRON, (e.g., +ها +hA *her/it/its*). Next comes the class of particles (PART+), (e.g., +س s+ *will [future]*). Most shallow is the class of conjunctions (CONJ+), (e.g., +و w+ *and*). We use the following five schemes: AR, $D1$, $D2$, $D3$ and TB. Definitions and contrastive examples of these schemes are presented in Table 1. To create these schemes, we use MADA, an off-the-shelf resource for Arabic morphological disambiguation (Habash and Rambow, 2005), and TOKAN, a general Arabic tokenizer (Habash and Sadat, 2006).

4 Preprocessing Schemes for Alignment

Using a preprocessing scheme for word alignment breaks the process of applying Giza++ (Och and Ney, 2003) on some parallel text into three steps: preprocessing, alignment and remapping. In preprocessing, the words are tokenized into smaller units. Then, they are passed along to Giza++ for alignment (default settings). Finally, the Giza++ alignments are mapped back (remapped) to the original word form which is AR tokens in this work. For instance, take the first word in Table 1, *wsyktbhA*; if the $D3$ preprocesssing scheme is applied to it before alignment, it is turned into four tokens (*w+ s+ yktb +hA*). Giza++ will link these tokens to different words on the English side. In the remapping step, the union of these links is assigned to the original word *wsyktbhA*. We refer to such alignments as remappings.

5 Alignment Combination

After creating the multiple remappings, we pass them as features into an alignment combiner. The combiner is also given a variety of additional features, which we discuss later in this section. The combiner is simply a binary classifier that determines for each source-target pair whether they are linked or not. Given the large size of the data used, we use a simplifying heuristic that allows us to minimize the number of source-target pairs used in training. Only links evidenced by at least one of the initial alignments and their immediate neighbors are included. All other links are considered non-existent. The combiner we use here is implemented using a rule-based classifier, Ripper (Cohen, 1996). The reasons we use Ripper as opposed other machine learning approaches are: (a) Ripper produces human readable rules that allow better understanding of the kind of decisions being made; and (b) Ripper is relatively fast compared to other machine learning approaches we examined given the very large nature of the training data we use. The combiner is trained using supervised data (human annotated alignments), which we discuss in Section 6.1.

In the rest of this section we describe the different machine learning features given to the combiner. We break the combination features in two types: word/sentence level and remapping features.

Word/Sentence Features:

• **Word Form**: The source and target word forms.

• **POS**: The source and target part-of-speech tags.

• **Location**: The source and target *relative* sentence position (the ratio of absolute position to sentence length). We also use the difference between these values for both source and target.

• **Frequency**: The source and target word frequency computed as the number of occurrences of the word form in training data. We also use the ratio of source to target frequency.

Similarity: This feature is motivated by the fact that proper nouns in different languages often resemble each other, e.g. صدام حسين 'SdAm Hsyn' and 'saddam hussein'. We use the equivalence classes proposed by Freeman et al. (2006) to normalize Arabic and English word forms. Then, we employ the longest common substring as a similarity measure.

Remapping Features:

• **Link**: for each source-target link, we include (a) a binary value indicating whether the link exists according to each remapping; (b) a cumulative sum of the different remappings supporting this link; and (c) co-occurrence information for this link. This last value is calculated for each source-target word pair as a weighted average of the product of the relative frequency of co-occurrence in both directions for each remapping. The weight assigned to each

remapping is computed empirically.[3]

• **Neighbor**: The same information as Link, but for each of the immediate neighbors of the current link.

• **Cross**: These include (a) the number of source words linked to the current target word, the same for target to source, and the number of words linked to either of the current words; and (b) the ratio of the co-occurrence mass placed in this link to the total mass assigned to the source word, the same for the target word and the union of both.

6 Evaluation

6.1 Experimental Data and Metrics

The gold standard alignments we use here are part of the IBM Arabic-English aligned corpus (IBMAC)[4] (Ittycheriah and Roukos, 2005). We only use 8.8K sentences from IBMAC because the rest (smaller portion) of the corpus uses different normalizations for numerals that make the two sets incompatible. We break this data into 6.6K sentences for training and 2.2K sentences for development. As for test data, we use the IBMAC's test set: NIST MTEval 2003 (663 Arabic sentences each human aligned to four English references).

To get initial Giza++ alignments, we use a larger parallel corpus together with the annotated set. The Arabic-English parallel corpus has about 5 million words.[5] The Arabic text in IBMAC is preprocessed in the AR preprocessing scheme with some additional character normalizations. We match the preprocessing and normalizations on our additional data to that of IBMAC's Arabic and English preprocessing (Ittycheriah and Roukos, 2005).

The standard evaluation metric within word alignment is the Alignment Error Rate (AER) (Och and Ney, 2000), which requires gold alignments that are marked as 'sure' or 'probable'. Since the IBMAC gold alignments we use are not marked as such, AER reduces to 1 - F-score (Ittycheriah and Roukos, 2005):

$$Pr = \frac{|A \cap S|}{|A|} \quad Rc = \frac{|A \cap S|}{|S|} \quad AER = 1 - \frac{2PrRc}{Pr+Rc}$$

where A links are proposed and S links are gold. NULL links are not included in the evaluation (Ayan, 2005; Ittycheriah and Roukos, 2005).

6.2 Results

We conducted three experiments on our development data: (a) to assess the contribution of alignment remapping, (b) to assess the contribution of combination features for a single alignment (i.e., independent of the combination task) and (c) to determine the best performing combination of alignment remappings. Experiments (b) and (c) used only 2.2K of the gold alignment training data to minimize computation time. As for our test data experiment, we use our best system with all of the available data. We also present an error analysis of our best system. The baseline we measure against in all of these experiments is the state-of-the-art grow-diag-final (*gdf*) alignment refinement heuristic commonly used in phrase-based SMT (Koehn et al., 2003). This heuristic adds links to the intersection of two asymmetrical statistical alignments in an attempt to assign every word a link. The AER of this baseline is 24.77%.

The Contribution of Alignment Remapping We experimented with five alignment remappings in two directions: *dir* (Ar-En) and *inv* (En-Ar). We also constructed their corresponding *gdf* alignment. The more verbose a preprocessing scheme, the lower the AER for either direction and for *gdf* of the corresponding remapping. The order of the schemes from worst to best is AR, $D1$, $D2$, TB and $D3$. The best result we obtained through remapping is that of $D3_{gdf}$ which had a 20.45% AER (17.4% relative decrease from the baseline).

The Contribution of Combination Features For each of the basic ten (non *gdf*) alignment remappings, we trained a version of the combiner that uses all the relevant features but has access to one alignment at a time. We saw a substantial improvement for all alignment remappings averaging 29.9% relative decrease in AER against the basic remapped version. The range of AER values is from 14.5% ($D3_{dir}$) to 20.79% (AR_{inv}).

Alignment Combination Experiments To determine the best subset of alignment remappings to combine, we ordered the alignments given their AER performance in the last experiment described (using combination features). Starting with the best performer ($D3_{dir}$), we continued adding alignments in the order of their performance so long the com-

[3] We use the AER on the development data normalized so all weights sum to one. See Section 6.2.

[4] We thank IBM for making their hand aligned data available to the research community.

[5] All of the training data we use is available from the Linguistic Data Consortium (LDC). The parallel text includes Arabic News, eTIRR, English translation of Arabic Treebank, and Ummah.

Table 2: Combining the Alignment Remappings

Alignment Remapping combination	AER
$D3_{dir}$	14.50
$D3_{dir}D2_{dir}$	14.12
$D3_{dir}D2_{dir}D3_{inv}$	12.81
$D3_{dir}D2_{dir}D3_{inv}D1_{dir}$	12.75
$D3_{dir}D2_{dir}D3_{inv}D1_{dir}AR_{inv}$	12.69

bination's AER score is decreased. Our best combination results are listed in Table 2. All additional alignments not listed in this table caused an increase in AER. The best alignment combination used alignments from four different schemes which confirms our intuition that such combination is useful.

Test Set Evaluation We ran our best system trained on all of the IBMAC data (training & development), on all the unseen IBMAC test set. On this data we achieve a substantial relative improvement of 38.3% from an AER of 22.99 to 14.19.

Ittycheriah and Roukos (2005) used only the top 50 sentences in IBMAC test data. Our best AER result on their test set is 14.02% (baseline is 22.48%) which is higher than their reported result (12.2% with 20.5% baseline (unrefined GIZA++)). The two results are not comparable because: (a) Ittycheriah and Roukos (2005) used additional gold aligned data that was not released and (b) they use an additional 500K sentences from the LDC UN corpus for Giza training that was created by adapting to the source side of the test set – the details of such adaptation were not provided and thus it is not clear how to replicate them to compare fairly. Clearly this additional data is helpful since even their baseline is higher than ours.[6]

Error Analysis We conducted error analysis on 50 sentences from our development set. The majority of the errors involved high frequency closed-class words (54%) and complex phrases (non-compositional or divergent translations) (23%). Both kinds of errors could be partly addressed by introducing phrasal constraints which are currently lacking in our system. Orthogonally, about 18% of all errors involved gold-standard inconsistencies and errors. These gold errors are split equally between closed-class and complex-phrase errors.

[6] Abraham Ittycheriah, personal communication.

7 Conclusion and Future Plans

We have presented an approach for using and combining multiple alignments created using different preprocessing schemes. We have shown a relative reduction of AER of about 38% on a blind test set. In the future, we plan to extend our system with additional models at the phrase and multi-word levels for both alignment and alignment combination improvement. We plan to use more sophisticated machine learning models such as support vector machines for combination and make use of more available parallel data. We also plan to evaluate the influence of our alignment improvement on MT quality.

References

N. Ayan. 2005. *Combining Linguistic and Machine Learning Techniques for Word Alignment Improvement*. Ph.D. thesis, University of Maryland, College Park.

W. Cohen. 1996. Learning trees and rules with set-valued features. In *Fourteenth Conference of the American Association of Artificial Intelligence*. AAAI.

A. Fraser and D. Marcu. 2006. Semi-supervised training for statistical word alignment. In *ACL-06*.

A. Freeman, S. Condon, and C. Ackerman. 2006. Cross linguistic name matching in English and Arabic. In *HLT-NAACL-06*.

N. Habash and O. Rambow. 2005. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *ACL-05*.

N. Habash and F. Sadat. 2006. Arabic Preprocessing Schemes for Statistical Machine Translation. In *HLT-NAACL-06*.

A. Ittycheriah and S. Roukos. 2005. A maximum entropy word aligner for arabic-english machine translation. In *EMNLP-05*.

P. Koehn, F. Och, and D. Marcu. 2003. Statistical Phrase-based Translation. In *HLT-NAACL-03*.

Y. Lee. 2004. Morphological Analysis for Statistical Machine Translation. In *HLT-NAACL-04*.

Y. Liu, Q. Liu, and S. Lin. 2005. Log-linear models for word alignment. In *ACL-05*.

A. Lopez and P. Resnik. 2006. Word-based alignment, phrase-based translation: what's the link? In *AMTA-06*.

F. Och and H. Ney. 2000. Improved statistical alignment models. In *ACL-2000*.

F. Och and H. Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–52.

F. Sadat and N. Habash. 2006. Combination of Arabic Preprocessing Schemes for Statistical Machine Translation. In *ACL-06*.

B. Taskar, S. Lacoste-Julien, and D. Klein. 2005. A discriminative matching approach to word alignment. In *EMNLP-05*.

C. Tillmann, S. Vogel, H. Ney, and A. Zubiaga. 1997. A DP-based search using monotone alignments in statistical translation. In *ACL-97*.

D. Yarowsky, G. Ngai, and R. Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *HLT-01*.

A Fast Method for Parallel Document Identification

Jessica Enright and **Grzegorz Kondrak**
Department of Computing Science
University of Alberta
Edmonton, AB, T6G 2E8, Canada
{enright,kondrak}@cs.ualberta.ca

Abstract

We present a fast method to identify homogeneous parallel documents. The method is based on collecting counts of identical low-frequency words between possibly parallel documents. The candidate with the most shared low-frequency words is selected as the parallel document. The method achieved 99.96% accuracy when tested on the EUROPARL corpus of parliamentary proceedings, failing only in anomalous cases of truncated or otherwise distorted documents. While other work has shown similar performance on this type of dataset, our approach presented here is faster and does not require training. Apart from proposing an efficient method for parallel document identification in a restricted domain, this paper furnishes evidence that parliamentary proceedings may be inappropriate for testing parallel document identification systems in general.

1 Introduction

Parallel documents are documents that are mutual translations. There are a number of reasons one might want to either identify parallel documents, or confirm that a pair of documents are in fact parallel. Most prominently, one could use pairs of automatically detected parallel documents to build parallel corpora. Parallel corpora have many uses in natural language processing, and their dearth has been identified as a major bottleneck (Diab, 2004). They have been employed in word sense disambiguation (Diab and Resnik, 2002), automatic construction of bilingual dictionaries (McEwan et al., 2002), and inducing statistical machine translation models (Koehn et al., 2003). In addition to building parallel corpora, one can envision other uses for parallel document identification, such as cross-language information retrieval (Chen and Nie, 2000).

Much work on identifying pairs of parallel documents focuses on the use of external features of the documents, rather than content. Chen and Nie (2000) describe PTMiner, a cross-language information retrieval system. They consider a number of factors in determining if a pair of documents are parallel, including document size, date, URL, and language flag. For example, if a document is available in both French and English, it is common for the French document's URL to contain $.fr$ and the English to contain $.en$ In addition to these measures, they consider website structure.

McEwan et al. (2002) find parallel documents which they then use to automatically build a bilingual dictionary. In their system, they first generate a set of candidate pairs based on manual selection, or advanced search engine use. They then filter the pairs to remove non-parallel pairs. First, they confirm that one of each pair is in each of the desired languages using tuned lists of stop-words, then they compare the documents based on length in tokens, and HTML markup. Resnik and Smith (2003) use a similar idea of candidates and filters in their STRAND system. STRAND filters the documents based on aligning them by length in tokens and location of HTML markup in the documents.

Apart form the work done on external metrics, Patry and Langlais (2005) investigated a number of content-based metrics. They consider several docu-

Proceedings of NAACL HLT 2007, Companion Volume, pages 29–32,
Rochester, NY, April 2007.

ment features, including the numbers, proper names and punctuation contained within, as well as document length, and alignment scores between candidate pairs. The features are then used to train an Ada-Boost classifier, which makes decisions based on edit-distance and cosine scores. They experimented with several combinations of features, one of which achieved 100% correctness when tested on 487 out of 488 parallel documents that constitute the English-Spanish portion of the EUROPARL corpus. They conclude that a bag-of-words approach is inferior to one that considers feature order.

In this work, we demonstrate that a much simpler approach can achieve equally good results. Our method does not depend on hand-coded linguistic knowledge and requires no training data, which may be unavailable for some language pairs. In addition, thanks to its simplicity, our method is very fast.

2 Parallel document identification

One can consider the parallel document identification problem to be as follows:

> Given one document d_A in language A, and a set of documents D_B in language B, identify exactly one document $d_B \in D_B$ that is the parallel, or translation, of d_A.

We initially designed a cognate-based approach to the problem, which employed a combination of orthographic word similarity measures to identify cognates such as French *nombres* and English *numbers* between documents. In order to make the method computationally feasible, potential cognates were filtered based on word order, location in the document, frequency, and length. However, we found that a faster and simpler procedure, which is described below, performed extremely well, eliminating the need for a more sophisticated approach.

We propose to identify parallel documents by counting the number of unique words that appear in both documents. The documents are treated as bags of words, that is, their word order is not considered. From each document, we extract a set of words that are at least 4 characters long and have frequency 1. Given a document in language A, we select the document in language B that shares the largest number of these words. An implementation based on hash tables ensures speed.

Since identical words of frequency 1 are almost certainly cognates, this method can be seen as an extremely conservative approach to cognate detection. In practice, most of unique identical words are proper nouns.

3 Experimental setup

We performed experiments on two different parliamentary corpora. The English-French Canadian Hansards from the 36th sitting of the Canadian Parliament (Germann, 2001) was selected as the development dataset. In testing on the Canadian Hansards, English was used as the Language A, and French as the Language B. Our approach correctly identified all parallel documents.

In order to allow for a direct comparison with the work of Patry and Langlais (2005), we adopted the EUROPARL corpus of parliamentary proceedings (Koehn, 2002) as our test dataset. However, rather than focusing on a single language pair, we performed tests on all 110 language pairs involving the following 11 languages: German, English, Greek, Finnish, Swedish, Dutch, French, Danish, Italian, Spanish and Portuguese. Diacritics were stripped from the documents of all languages. Since Greek utilizes a different script from the rest of the documents. we used a straightforward context-free mapping to convert every Greek character to its nearest roman equivalent.

Some of the 488 documents available in EUROPARL were missing in Finnish, Swedish, Greek and Danish. In particular, Greek had 392 documents, Danish had 487 documents, and Swedish and Finnish had 433 each. In such cases, the parallels of those missing documents were excluded from the language A for that test.

The EUROPARL documents range in size from 114 tokens (13 lines) to 138,557 tokens (11,101 lines). The mean number of tokens is 59,387 (2,826 lines). Each orientation of each language pair was tested. For example, for the language pair English-Dutch, tests were run twice - once with English as language A and Dutch as language B, and once the other way around. The results for a given language pair are not necessarily symmetric. Henceforth when referring to a language pair, we list the language A as the first one.

For each document and each language pair, an individual test was run. An individual test consisted of finding, for a given document in language A, its parallel in the language B set. Since we did not take advantage of the pigeon-hole constraint, the individual tests were independent from each other.

No changes were made to the approach once testing on the EUROPARL corpus began, in order to avoid adapting it to work on any particular data set.

4 Results

In total, only 20 of the 49872 tests did not produce the correct result (0.04% error rate). There was one incorrect selection in the English-Spanish language pair, one in the English-German pair, as well as in each of 18 language pairs involving Danish or English as a Language A. All of the incorrect results can be traced to mistranslation, or to missing/truncated documents. In particular, one of the documents is severely truncated in Danish and English, one of the German documents missing a portion of its text, and the Spanish version of one of the documents contains a number of phrases and sentences of English, apparently belonging to the English version of the text.

Effectively, when this method fails it is because the input does not match the problem definition. Recall that the problem was defined as selecting a document d_B from a set of documents D_B in language B that is the correct parallel to d_A, a document in language A. Failure cases occurred because there was no correct parallel to the d_A in D_B. In fact, each of the "incorrect" results is a manifestation of an editorial error in the EUROPARL corpus. One could see this approach being used as an aid to identifying fragmentary documents and mistranslations in parallel corpora.

Encouraged by the excellent accuracy of our method, we decided to try an even simpler approach, which is based on words of frequency 1 in the entire set of documents in a given language, rather than in a single document. For every document from a language A, we select as its parallel the document from language B that shares the most of those words with it. However, the results obtained with this method were clearly inferior, with the error rates ranging from 2.9% for Dutch to 27.3% for Finnish.

5 Discussion

The implications of this work are two-fold. First, it shows a simple, fast, and effective method for identifying parallel documents. Second, it calls into question the usefulness of parliamentary proceedings for the evaluation of parallel document identification schemes.

The method described in this paper is sufficiently simple as to be used as a baseline for comparison with other methods. No information is shared between trials, no word similarity measures are used, and word order is ignored. The method does not incorporate any language-specific linguistic knowledge, and it has shown itself to be robust across languages without any alterations. The only constraint is that the languages must share an alphabet, or can be converted into a common alphabet. Furthermore, it requires no training phase, which would likely have to be repeated for every pair of languages.

Our method achieves 99.9% accuracy on the English-Spanish language pair, which roughly matches the best result reported by Patry and Langlais (2005) (who apparently removed one document pair from the collection). However, their method requires a training phase on aligned parallel documents, making it time consuming and inconvenient to adapt their approach to a new language pair, even in cases where such document-aligned corpora are available. In addition, their top accuracy value corresponds to only one of several combination of features — the results with classifiers based on other combinations of features were lower.

We implemented our method using hash tables, which store the words occurring in a document together with their frequencies. This makes the entire search for a parallel document roughly linear in the total number of words in all the documents. Average total wall-clock time spent for one test with one language A document and 488 language B documents was 59.4 seconds. on a AMD Athlon(tm) 64 Processor 3500+. Profiling showed that on average 99.7% of the wall-clock time was spent on I/O operations, with the remainder taken by hash table lookups and string equality checks. Clearly, little speed improvement is possible. In contrast to the speed of our approach, the approach used by Patry and Langlais (2005) requires not only the time to train a classifier,

but also the time to compute edit distance between many document pairs.

In addition to yielding a simple, accurate and fast method for parallel document identification, our results suggest that relatively "clean" collections of parliamentary proceedings of the EUROPARL type may be inappropriate for testing parallel document identification schemes in general. If a very simple approach can achieve near perfect accuracy in such a domain, perhaps the task is too easy. Future general parallel document identification systems should be tested on more challenging datasets.

6 Future Work

While the approach presented here has been very successful thus far, there are a number of extensions that could be made to make it more applicable in general. More work could allow it to deal with cases of missing parallel documents, datasets with fewer proper names, and even yield knowledge of the difficulty of the problem in general.

First, the problem definition could be expanded to include cases where there is no valid parallel for a given language A document in the language B document set. This could take the form of establishing a score or significance threshold. For example, if there were no document in the language B set that shared more than the minimum number of unique words with the document d_A in language A, then the approach might return no parallel for that document.

Second, it might be revealing to run further tests with this approach on other types of text than parliamentary proceedings. What types of text would require a more sophisticated approach? The answer to that question might have implications for the range of text types that ought to be used to comprehensively test parallel document identification systems.

The exact matching of words is a critical feature of our approach, which enables it to perform quick comparisons of documents by representing them as sets of low-frequency words stored in hash tables. However, it is also a limitation because many cross-language cognates are not orthographically identical. A system relying on non-binary word similarity measures rather than on total identity of words would be more complex and slower, but also more robust across different domains of text.

7 Conclusion

We have presented a viable, simple method for identification of homogeneous parallel documents. This method uses less resources and time than other content-based methods, a valuable asset when many languages lack linguistic resources. In addition to showing the effectiveness of our approach, the results of the experiments suggest that parliamentary proceedings may be inappropriate for parallel document identification scheme testing.

Acknowledgments

We would like to thank Colin Cherry and other members of the NLP research group at University of Alberta for their helpful comments and suggestions. This research was supported by the Natural Sciences and Engineering Research Council of Canada.

References

Jiang Chen and Jian-Yun Nie. 2000. Parallel web text mining for cross-language IR. In *In Proc. of RIAO*, pages 62–77.

Mona Diab and Philip Resnik. 2002. An unsupervised method for word sense tagging using parallel corpora. In *Proc. of ACL*, pages 255–262.

Mona Diab. 2004. Relieving the data acquisition bottleneck for word sense disambiguation. In *Proc. of ACL*, pages 303–310.

Ulrich Germann. 2001. Aligned Hansards of the 36th Parliament of Canada, Release 2001-1a. Available at http://www.isi.edu/natural-language/download/hansard/.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL*, pages 48–54.

Philipp Koehn. 2002. Europarl: A multilingual corpus for evaluation of machine translation. Available at http://people.csail.mit.edu/koehn/.

Craig J. A. McEwan, Iadh Ounis, and Ian Ruthven. 2002. Building bilingual dictionaries from parallel web documents. In *Proc. of ECIR*, pages 303–323.

Alexandre Patry and Philippe Langlais. 2005. Automatic identification of parallel documents with light or without linguistic resources. In *Proc. of Canadian AI*, pages 354–365.

Philip Resnik and Noah A. Smith. 2003. The web as a parallel corpus. *Comput. Linguist.*, 29(3):349–380.

Generalized Graphical Abstractions for Statistical Machine Translation

Karim Filali and **Jeff Bilmes***
Departments of Computer Science & Engineering and Electrical Engineering
University of Washington
Seattle, WA 98195, USA
{karim@cs,bilmes@ee}.washington.edu

Abstract

We introduce a novel framework for the expression, rapid-prototyping, and evaluation of statistical machine-translation (MT) systems using graphical models. The framework extends dynamic Bayesian networks with multiple connected different-length streams, switching variable existence and dependence mechanisms, and constraint factors. We have implemented a new general-purpose MT training/decoding system in this framework, and have tested this on a variety of existing MT models (including the 4 IBM models), and some novel ones as well, all using Europarl as a test corpus. We describe the semantics of our representation, and present preliminary evaluations, showing that it is possible to prototype novel MT ideas in a short amount of time.

1 Introduction

We present a unified graphical model framework based on (Filali and Bilmes, 2006) for statistical machine translation. Graphical models utilize graphical descriptions of probabilistic processes, and are capable of quickly describing a wide variety of different sets of model assumptions. In our approach, either phrases or words can be used as the unit of translation, but as a first step, we have only implemented word-based models since our main goal is to show the viability of our graphical model representation and new software system.

There are several important advantages to a unified probabilistic framework for MT including: **(1)** the same codebase can be used for training and decoding without having to implement a separate decoder for each model; **(2)** new models can be prototyped quickly; **(3)** combining models (such as in a speech-MT system) is easier when they are encoded in the same framework; **(4)** sharing algorithms across different disciplines (e.g., the MT and the constraint-satisfaction community) is facilitated.

2 Graphical Model Framework

A *Graphical Model* (GM) represents a factorization of a family of joint probability distributions over a set of random variables using a graph. The graph specifies conditional independence relationships between the variables, and parameters of the model are associated with each variable or group thereof. There are many types of graphical models. For example, Bayesian networks use an acyclic directed graph and their parameters are conditional probabilities of each variable given its parents. Various forms of GM and their conditional independence properties are defined in (Lauritzen, 1996).

Our graphical representation, which we call *Multi-dynamic Bayesian Networks* (MDBNs) (Filali and Bilmes, 2006), is a generalization of dynamic Bayesian networks (DBNs) (Dean and Kanazawa, 1988). DBNs are an appropriate representation for sequential (for example, temporal) stochastic processes, but can be very difficult to apply when dependencies have arbitrary time-span and the existence of random variables is contingent on the val-

*This material was supported by NSF under Grant No. ISS-0326276.

Proceedings of NAACL HLT 2007, Companion Volume, pages 33–36,
Rochester, NY, April 2007.

ues of certain variables in the network. In (Filali and Bilmes, 2006), we discuss inference and learning in MDBNs. Here we focus on representation and the evaluation of our new implementation and framework. Below, we summarize key features underlying our framework. In section 3, we explain how these features apply to a specific MT model.

- Multiple DBNs can be represented along with rules for how they are interconnected — the rule description lengths are fixed (they do not grow with the length of the DBNs).
- *Switching dependencies* (Geiger and Heckerman, 1996): a variable X is a *switching parent* of Y if X influences what type of dependencies Y has with respect to its other parents, e.g., an *alignment* variable in IBM Models 1 and 2 is switching.
- *Switching existence*: A variable X is "switching existence" with respect to variable Y if the value of X determines whether Y exists. An example is a *fertility* variable in IBM Models 3 and above.
- Constraints and aggregation: BN semantics can encode various types of constraints between groups of variables (Pearl, 1988). For example, in the construct $A \rightarrow B \leftarrow C$ where B is observed, B can constrain A and C to be unequal. We extend those semantics to support a more efficient evaluation of constraints under some variable order conditions.

3 GM Representation of IBM MT Models

In this section we present a GM representation for IBM model 3 (Brown et al., 1993) in fig. 1. Model 3 is intricate enough to showcase some of the features of our graphical representation but not as complex as, and thus is easier to describe, than model 4. Our choice of representing IBM models is not because we believe they are state of the art MT models—although they are still widely used in producing alignments and as features in log-linear models—but because they provide a good initial testbed for our architecture.

The topmost random variable (RV), ℓ, is a hidden switching existence variable corresponding to the length of the English string. The box abutting ℓ includes all the nodes whose existence depends on the value of ℓ. In the figure, $\ell = 3$, thus resulting in three English words $e_1, ..., e_3$, connected using a second-order Markov chain. To each English word e_i corresponds a conditionally dependent fertility ϕ_i, which indicates how many times e_i is used by words in the French string. Each ϕ_i in turn grants existence to a set of RVs under it. Given the fertilities (the figure depicts the case $\phi_1 = 3, \phi_2 = 1, \phi_3 = 0$), for each word e_i, ϕ_i French word RVs are granted existence and are denoted by the *tablet* $\tau_{i1}, \tau_{i2}, \ldots, \tau_{i\phi_i}$ of e_i. The values of τ variables need to match the actual observed French sequence $f_1, \ldots, f_m$. This is represented as a shared constraint between all the f, π, and τ variables which have incoming edges into the observed variable v. v's conditional probability table is such that it is one only when the associated constraint is satisfied. The variable $\pi_{i,k}$ is a switching dependency parent with respect to the constraint variable v and determines which f_j participates in an equality constraint with $\tau_{i,k}$.

In the null word sub-model, the constraint that successive permutation variables be ordered is implemented using the observed child w of π_{0i} and $\pi_{0(i+1)}$. The probability of w being unity is one only when the constraint is satisfied and zero otherwise.

The bottom variable m is a switching existence node (observed to be 6 in the figure) with corresponding French word sequence and alignment variables. The French sequence participates in the v constraint described above, while the alignment variables $a_j \in \{1, \ldots, \ell\}, j \in 1, \ldots, m$ constrain the fertilities to take their unique allowable values (for the given alignment). Alignments also restrict the domain of permutation variables, $\boldsymbol{\pi}$, using the constraint variable x. Finally, the domain size of each a_j has to lie in the interval $[0, \ell]$ and that is enforced by the variable u. The dashed edges connecting the alignment a variables represent an extension to implement an M3/M-HMM hybrid.[1]

4 Experiments

We have developed (in C++) a new entirely self-contained general-purpose MT training/decoding system based on our framework, of which we provide a preliminary evaluation in this section. Although the framework is perfectly capable of representing phrase-based models, we restrict ourselves to word-based models to show the viability of graphical models for MT and will consider different translation units in future work. We perform MT ex-

[1] We refer to the HMM MT model in (Vogel et al., 1996) as M-HMM to avoid any confusion.

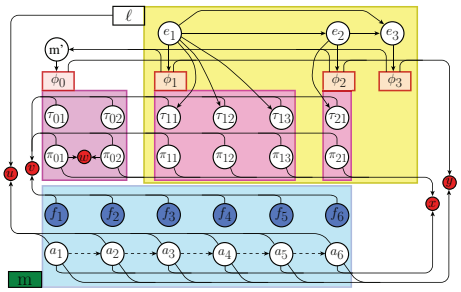


Figure 1: Unrolled Model 3 graphical model with fertility assignment $\phi_0 = 2, \phi_1 = 3, \phi_2 = 1, \phi_3 = 0$.

periments on a English-French subset of the Europarl corpus used for the ACL 2005 SMT evaluations (Koehn and Monz, 2005). We train an English language model on the whole training set using the SRILM toolkit (Stolcke, 2002) and train MT models mainly on a 10k sentence pair subset of the ACL training set. We test on the 2000 sentence test set used for the same evaluations. For comparison, we use the MT training program, GIZA++ (Och and Ney, 2003), the phrase-base decoder, Pharaoh (Koehn et al., 2003), and the word-based decoder, Rewrite (Germann, 2003).

For inference we use a backtracking depth-first search inference method with memoization that extends Value Elimination (Bacchus et al., 2003). The same inference engine is used for both training and decoding. As an admissible heuristic for decoding, we compute, for each node V with Conditional Probability Table c, the largest value of c over all possible configurations of V and its parents (Filali and Bilmes, 2006).

Decoder	**BLEU (%)**			
	500	**1000**	**1500**	**2000**
Rewrite	25.3	22.3	21.7	22.01
Pharaoh	20.4	18.1	17.7	18.05
M-HMM	19.9	16.9	15.6	12.5

Table 1: *BLEU scores on first 500, 1000, 1500, and 2000 sentences (ordered from shortest to longest) of the ACL05 English-French 2000 sentence test set using a 700k sent train set. The last row is our MDBN system's simulation of a M-HMM model.*

Table 1 compares MT performance between (1) Pharaoh (which uses beam search), (2) our system, and (3) Rewrite (hill-climbing). (1) and (2) make use of a fixed lexical table[2] learned using an M-HMM model specified using our tool, and neither uses minimum error rate training. (3) uses Model 4 parameters learned using GIZA++. This comparison is informative because Rewrite is a special purpose model 4 decoder and we would expect it to perform at least as well as decoders not written for a specific IBM model. Pharaoh is more general in that it only requires, as input, a lexical table from any given model.[3] Our MDBN system is not tailored for the translation task. Pharaoh was able to decode the 2000 sentences of the test set in 5000s on a 3.2GHz machine; Rewrite took 84000s, and we allotted 400000s for our engine (200s per sentence). We attribute the difference in speed and BLEU score between our system and Pharaoh to the fact Value Elimination searches in a depth-first fashion over the space of *partial configurations of RVs*, while Pharaoh expands *partial translation hypotheses* in a best-first search manner. Thus, Pharaoh can take advantage of knowledge about the MT problem's hypothesis space while the GM is agnostic with respect to the structure of the problem—something that is desirable from our perspective since generality is a main concern of ours. Moreover, the MDBN's heuristic and caching of previously explored subtrees have not yet proven able to defray the cost, associated with depth-first search, of exploring subtrees that do not contain any "good" configurations.

Table 2 shows BLEU scores of different MT models trained using our system. We decode using Pharaoh because the above speed difference in its favor allowed us to run more experiments and focus on the training aspect of different models. **M1, M2, M-HMM, M3, and M4** are the standard IBM models. **M2d** and **M-Hd** are variants in which the distortion between the French and English positions is used instead of the absolute alignment position. **M-Hdd** is a second-order **M-HMM** model (with distortion). **M3H** (see fig 1) is a variant of model 3 that uses first-order dependencies between alignment variables. **M-Hr** is another HMM model that uses the relative distortion between the current alignment and the previous one. This is similar to the model implemented by GIZA except we did

[2] Pharaoh's phrases are single words only.

[3] It does, however, use simple hard-coded distortion and fertility models.

	BLEU(%)			
	Giza train		MDBN train	
	10k	700k	10k	700k
M1	15.67	18.04	14.53	17.74
M2	15.84	18.52	15.74	
M2d	NA	NA	15.75	
M-HMM	NA	NA	15.87	
M-Hd	NA	NA	15.99	18.05
M-Hdd	NA	NA	15.55	
M-Hr	16.98	19.57	16.04	
M3	16.78	19.38	15.32	
M3H	NA	NA	15.67	
M4	16.81	19.51	15.00	
M4H	NA	NA	15.20	

Table 2: *BLEU scores for various models trained using GM and GIZA (when applicable). All models are decoded using Pharaoh.*

not include the English word class dependency. Finally, model **M4H** is a simplified model 4, in which only distortions within each tablet are modeled but a Markov dependency is also used between the alignment variables.

Table 2 also shows BLEU scores obtained by training equivalent IBM models using GIZA and the standard training regimen of initializing higher models with lower ones (we use the same schedules for our GM training, but only transfer lexical tables). The main observation is that GIZA-trained M-HMM, M3 and 4 have about 1% better BLEU scores than their corresponding MDBN versions. We attribute the difference in M3/4 scores to the fact we use a Viterbi-like training procedure (i.e., we consider a single configuration of the hidden variables in EM training) while GIZA uses pegging (Brown et al., 1993) to sum over a set of likely hidden variable configurations in EM.

While these preliminary results do not show improved MT performance, nor would we expect them to since they are on simulated IBM models, we find very promising the fact that this general-purpose graphical model-based system produces competitive MT results on a computationally challenging task.

5 Conclusion and Future Work

We have described a new probabilistic framework for doing statistical machine translation. We have focused so far on word-based translation. In future work, we intend to implement phrase-based MT models. We also plan to design better approximate inference strategies for training highly connected graphs such as IBM models 3 and 4, and some novel extensions. We are also working on new best-first search generalizations of our depth-first search inference to improve decoding time. As there has been increased interest in end-to-end task such as speech translation, dialog systems, and multilingual search, a new challenge is how best to combine the complex components of these systems into one framework. We believe that, in addition to the finite-state transducer approach, a graphical model framework such as ours would be well suited for this scientific and engineering endeavor.

References

F. Bacchus, S. Dalmao, and T. Pitassi. 2003. Value elimination: Bayesian inference via backtracking search. In *UAI-03*, pages 20–28, San Francisco, CA. Morgan Kaufmann.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

T. Dean and K. Kanazawa. 1988. Probabilistic temporal reasoning. *AAAI*, pages 524–528.

Karim Filali and Jeff Bilmes. 2006. Multi-dynamic Bayesian networks. In *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA.

Dan Geiger and David Heckerman. 1996. Knowledge representation and inference in similarity networks and bayesian multinets. *Artif. Intell.*, 82(1-2):45–74.

Ulrich Germann. 2003. Greedy decoding for statistical Machine Translation in almost linear time. In *HLT-NAACL*, pages 72–79, Edmonton, Canada, May. ACL.

P. Koehn and C. Monz. 2005. Shared task: Statistical machine translation between European languages. pages 119–124, Ann Arbor, MI, June. ACL.

P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *HLT-NAACL*, May.

S.L. Lauritzen. 1996. *Graphical Models*. Oxford Science Publications.

F. J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

J. Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

A. Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proc. Int. Conf. on Spoken Language Processing*.

S. Vogel, H. Ney, and C. Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING*, pages 836–841.

Situated Models of Meaning for Sports Video Retrieval

Michael Fleischman
MIT Media Lab
mbf@mit.edu

Deb Roy
MIT Media Lab
dkroy@media.mit.edu

Abstract

Situated models of meaning ground words in the non-linguistic context, or situation, to which they refer. Applying such models to sports video retrieval requires learning appropriate representations for complex events. We propose a method that uses data mining to discover temporal patterns in video, and pair these patterns with associated closed captioning text. This paired corpus is used to train a situated model of meaning that significantly improves video retrieval performance.

1 Introduction

Recent advances in digital broadcasting and recording allow fans access to an unprecedented amount of sports video. The growing need to manage and search large video collections presents a challenge to traditional information retrieval (IR) technologies. Such methods cannot be directly applied to video data, even when closed caption transcripts are available; for, unlike text documents, the occurrence of a query term in a video is often not enough to assume the video's relevance to that query. For example, when searching through video of baseball games, returning all clips in which the phrase "home run" occurs, results primarily in video of events where a home run does not actually occur. This follows from the fact that in sports, as in life, people often talk not about what is currently happening, but rather, they talk about what did, might, or will happen in the future.

Traditional IR techniques cannot address such problems because they model the meaning of a query term strictly by that term's relationship to other terms. To build systems that successfully search video, IR techniques should be extended to exploit not just linguistic information but also elements of the non-linguistic context, or *situation*, that surrounds language use. This paper presents a method for video event retrieval from broadcast sports that achieves this by learning a *situated* model of meaning from an unlabeled video corpus.

The framework for the current model is derived from previous work on computational models of verb learning (Fleischman & Roy, 2005). In this earlier work, meaning is defined by a probabilistic mapping between words and representations of the non-linguistic events to which those words refer. In applying this framework to events in video, we follow recent work on video surveillance in which complex events are represented as temporal relations between lower level sub-events (Hongen et al., 2004). While in the surveillance domain, hand crafted event representations have been used successfully, the greater variability of content in broadcast sports demands an automatic method for designing event representations.

The primary focus of this paper is to present a method for mining such representations from large video corpora, and to describe how these representations can be mapped to natural language. We focus on a pilot dataset of broadcast baseball games. Pilot video retrieval tests show that using a situated model significantly improves performances over traditional language modeling methods.

2 Situated Models of Meaning

Building situated models of meaning operates in three phases (see Figure 1): first, raw video data is abstracted into multiple streams of discrete features. Temporal data mining techniques are then applied to these feature streams to discover hierarchical temporal patterns. These temporal patterns form the event representations that are then mapped to words from the closed caption stream.

2.1 Feature Extraction

The first step in representing events in video is to abstract the very high dimensional raw video data into more semantically meaningful streams of information. Ideally, these streams would correspond to basic events that occur in sports video (e.g., hitting, throwing, catching, kicking, etc.). Due to the limitations of computer vision techniques, extracting such ideal features is often infeasible. However, by exploiting the "language of

Proceedings of NAACL HLT 2007, Companion Volume, pages 37–40,
Rochester, NY, April 2007.

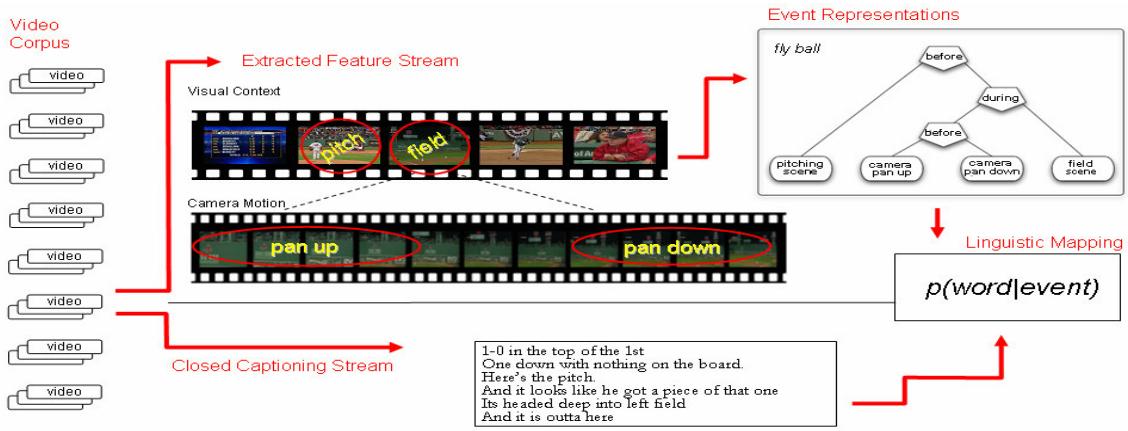


Figure 1. Video processing pipeline for learning situated models of meaning.

film" that is used to produce sports video, informative features can be extracted that are also easy to compute. Thus, although we cannot easily identify a player hitting the ball, we can easily detect features that correlate with hitting: e.g., when a scene focusing on the pitching mound immediately jumps to one zooming in on the field (Figure 1). While such correlations are not perfect, pilot tests show that baseball events can be classified using such features (Fleischman et. al., in prep).

Importantly, this is the only phase of our framework that is domain specific; i.e., it is the only aspect of the framework designed specifically for use with baseball data. Although many feature types can be extracted, we focus on only two feature types: visual context, and camera motion.

Visual Context

Visual context features encode general properties of the visual scene in a video segment. The first step in extracting such features is to split the raw video into "shots" based on changes in the visual scene due to editing (e.g., jumping from a close up of the pitcher to a wide angle of the field). Shot detection is a well studied problem in multimedia research; in this work, we use the method of Tardini et al. (2005) because of its speed and proven performance on sports video.

After a game is segmented into shots, each shot is categorized into one of three categories: *pitching-scene, field-scene*, or *other*. Categorization is based on image features (e.g., color histograms, edge detection, motion analysis) extracted from an individual key frame chosen from that shot. A decision tree is trained (with bagging and boosting) using the WEKA machine learning toolkit that achieves over 97% accuracy on a held out dataset.

Camera Motion

Whereas visual context features provide information about the global situation that is being observed, camera motion features afford more precise information about the actions occurring in the video. The intuition here is that the camera is a stand in for a viewer's focus of attention. As action in the video takes place, the camera moves to follow it, mirroring the action itself, and providing an informative feature for event representation.

Detecting camera motion (i.e., pan/tilt/zoom) is a well-studied problem in video analysis. We use the system of (Bouthemy et al., 1999) which computes the pan, tilt, and zoom motions using the parameters of a two-dimensional affine model fit to every pair of sequential frames in a video segment. The output of this system is then clustered into characteristic camera motions (e.g. zooming in fast while panning slightly left) using a 1st order Hidden Markov Model with 15 states, implemented using the Graphical Modeling Toolkit (GMTK).

2.2 Temporal Pattern Mining

In this step, temporal patterns are mined from the features abstracted from the raw video data. As described above, ideal semantic features (such as hitting and catching) cannot be extracted easily from video. We hypothesize that finding temporal patterns between scene and camera motion features can produce representations that are highly correlated with sports events. Importantly, such temporal patterns are not strictly sequential, but rather, are composed of features that can occur in complex and varied temporal relations to each other. For example, Figure 1 shows the representation for a fly ball event that is composed of: a *camera pan-*

ning up followed by a *camera pan down*, occurring during a *field scene,* and before a *pitching scene*.

Following previous work in video content classification (Fleischman et al., 2006), we use techniques from temporal data mining to discover event patterns from feature streams. The algorithm we use is fully unsupervised. It processes feature streams by examining the relations that occur between individual features within a moving time window. Following Allen (1984), any two features that occur within this window must be in one of seven temporal relations with each other (e.g. *before, during, etc.*). The algorithm keeps track of how often each of these relations is observed, and after the entire video corpus is analyzed, uses chi-square analyses to determine which relations are significant. The algorithm iterates through the data, and relations between individual features that are found significant in one iteration (e.g. [BEFORE, *camera panning up, camera panning down*]), are themselves treated as individual features in the next. This allows the system to build up higher-order nested relations in each iteration (e.g. [DURING, [BEFORE, *camera panning up, camera panning down*], *field scene*]]). The temporal patterns found significant in this way are then used as the event representations that are then mapped to words.

2.3 Linguistic Mapping

The last step in building a situated model of meaning is to map words onto the representations of events mined from the raw video. We equate the learning of this mapping to the problem of estimating the conditional probability distribution of a word given a video event representation. Similar to work in image retrieval (Barnard et al., 2003), we cast the problem in terms of Machine Translation: given a paired corpus of words and a set of video event representations to which they refer, we make the IBM Model 1 assumption and use the expectation-maximization method to estimate the parameters (Brown et al., 1993):

$$p(word \mid video) = \frac{C}{(l+1)^m} \prod_{j=1}^{m} p(word_j \mid video_{a_j}) \qquad \textbf{(1)}$$

This paired corpus is created from a corpus of raw video by first abstracting each video into the feature streams described above. For every shot classified as a *pitching scene*, a new instance is created in the paired corpus corresponding to an event that starts at the beginning of that shot and ends exactly four shots after. This definition of an event follows from the fact that most events in baseball must start with a pitch and usually do not last longer than four shots (Gong et al., 2004).

For each of these events in the paired corpus, a representation of the video is generated by matching all patterns (and the nested sub-patterns) found from temporal mining to the feature streams of the event. These video representations are then paired with all the words from the closed captioning that occur during that event (plus/minus 10 seconds).

3 Experiments

Work on video IR in the news domain often focuses on indexing video data using a set of image classifiers that categorize shots into pre-determined concepts (e.g. *flag*, *outdoors*, *George Bush, etc.).* Text queries must then be translated (sometimes manually) in terms of these concepts (Worring & Snoek, 2006). Our work focuses on a more automated approach that is closer to traditional IR techniques. Our framework extends the language modeling approach of Ponte and Croft (1998) by incorporating a situated model of meaning.

In Ponte and Croft (1998), documents relevant to a query are ranked based on the probability that each document generated each query term. We follow this approach for video events, making the assumption that the relevance of an event to a query depends both on the words associated with the event (i.e. what was said while the event occurred), as well as the situational context modeled by the video event representations:

$$p(query \mid event) = \prod_{word}^{query} p(word \mid caption)^{\alpha} * p(word \mid video)^{(1-\alpha)} \qquad \textbf{(2)}$$

The *p(word|caption)* is estimated using the language modeling technique described in Ponte and Croft (1998). The *p(word|video)* is estimated as in equation 1 above. α is used to weight the models.

Data

The system has been evaluated on a pilot set of 6 broadcast baseball games totaling about 15 hours and 1200 distinct events. The data represents video of 9 different teams, at 4 different stadiums, broadcast on 4 different stations. Highlights (i.e., events which terminate with the player either *out* or *safe*) were hand annotated, and categorized according to the type of the event (e.g., *strikeout vs. homerun)*, the location of the event (*e.g., right field vs. infield*), and the nature of the event (e.g., *fly ball vs. line drive*). Each of these categories was

used to automatically select query terms to be used in testing. Similar to Berger & Lafferty (1999), the probability distribution of terms given a category is estimated using a normalized log-likelihood ratio (Moore, 2004), and query terms are sampled randomly from this distribution. This gives us a set of queries for each annotated category (e.g., *strikeout*: "miss, chasing"; *flyball:* "fly, streak"). Although much noisier than human produced queries, this procedure generates a large amount of test queries for which relevant results can easily be determined (e.g., if a returned event for the query "fly, streak" is of the *flyball* category, it is marked relevant).

Experiments are reported using 6-fold cross validation during which five games are used to train the situated model while the sixth is held out for testing. Because data is sparse, the situation model is trained only on the hand annotated highlight events. However, retrieval is always tested using both highlight and non-highlight events.

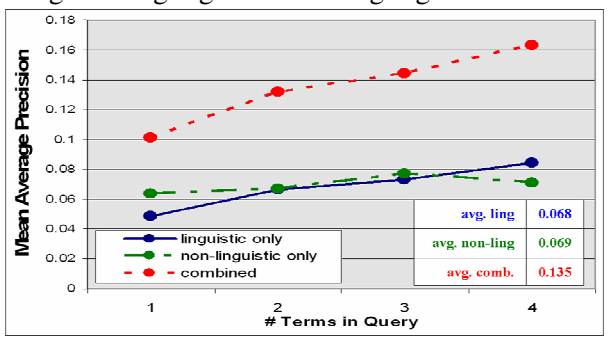


Figure 2. Effect of situated model on video IR.

Results

Figure 2 shows results for 520 automatically generated queries of one to four words in length. Mean average precision (MAP), a common metric that combines elements of precision, recall, and ranking, is used to measure the relevance of the top five results returned for each query. We show results for the system using only linguistic information (i.e. α=1), only non-linguistic information (i.e. α=0), and both information together (i.e. α=0.5).

The poor performance of the system using only non-linguistic information is expected given the limited training data and the simple features used to represent events. Interestingly, using only linguistic information produces similarly poor performance. This is a direct result of announcers' tendency to discuss topics not currently occurring in the video. By combining text and video analyses, though, the system performs significantly better ($p<0.01$) by determining when the observed language actually refers to the situation at hand.

4 Conclusion

We have presented a framework for video retrieval that significantly out-performs traditional IR methods applied to closed caption text. Our new approach incorporates the visual content of baseball video using automatically learned event representations to model the situated meaning of words. Results indicate that integration of situational context dramatically improves performance over traditional methods alone. In future work we will examine the effects of applying situated models of meaning to other tasks (e.g., machine translation).

References

Allen, J.F. (1984). A General Model of Action and Time. Artificial Intelligence. 23(2).

Barnard, K, Duygulu, P, de Freitas, N, Forsyth, D, Blei, D, and Jordan, M. (2003), "Matching Words and Pictures," Journal of Machine Learning Research, Vol 3.

Berger, A. and Lafferty, J. (1999). Information Retrieval as Statistical Translation. In Proceedings of SIGIR-99.

Bouthemy, P., Gelgon, M., Ganansia, F. (1999). A unified approach to shot change detection and camera motion characterization. IEEE Trans. on Circuits and Systems for Video Technology, 9(7):1030-1044.

Brown, P., Della Pietra, S., Della Pietra, V. Mercer, R. (1993). The mathematics of machine translation: Parameter estimation. Computational Linguistics, 19(10).

Fleischman, M. and Roy, D. (2005). Intentional Context in Situated Language Learning. In Proc. of 9th Conference on Comp. Natural Language Learning.

Fleischman, M., DeCamp, P. Roy, D. (2006). Mining Temporal Patterns of Movement for Video Content Classification. The 8th ACM SIGMM International Workshop on Multimedia Information Retrieval.

Fleischman, M., Roy, B., Roy, D. (in prep.). Automated Feature Engineering inBaseball Highlight Classification.

Gong, Y., Han, M., Hua, W., Xu, W. (2004). Maximum entropy model-based baseball highlight detection and classification. Computer Vision and Image Understanding. 96(2).

Hongen, S., Nevatia, R. Bremond, F. (2004). Video-based event recognition: activity representation and probabilistic recognition methods. Computer Vision and Image Understanding. 96(2). pp: 129 - 162

Moore, Robert C. (2004). Improving IBM Word Alignment Model 1. in Proc. of 42nd ACL.

Ponte, J.M., and Croft, W.B. (1998). A Language Modeling Approach to Information Retrieval. In Proc. of SIGIR'98.

Tardini, G. Grana C., Marchi, R., Cucchiara, R., (2005). Shot Detection and Motion Analysis for Automatic MPEG-7 Annotation of Sports Videos. In 13th International Conference on Image Analysis and Processing.

Worring, M., Snoek, C.. (2006). Semantic Indexing and Retrieval of Video. Tutorial at ACM Multimedia

Exploring Affect-Context Dependencies for Adaptive System Development

Kate Forbes-Riley
Learning R&D Ctr.
Univ. Pittsburgh
Pittsburgh, PA 15260
forbesk@pitt.edu

Mihai Rotaru
Computer Science Dpt.
Univ. Pittsburgh
Pittsburgh, PA 15260
mrotaru@cs.pitt.edu

Diane J. Litman
Learning R&D Ctr.
Computer Science Dpt.
Univ. Pittsburgh
Pittsburgh, PA 15260
litman@cs.pitt.edu

Joel Tetreault
Learning R&D Ctr.
Univ. Pittsburgh
Pittsburgh, PA 15260
tetreaul@pitt.edu

Abstract

We use χ^2 to investigate the context dependency of student affect in our computer tutoring dialogues, targeting uncertainty in student answers in 3 automatically monitorable contexts. Our results show significant dependencies between uncertain answers and specific contexts. Identification and analysis of these dependencies is our first step in developing an adaptive version of our dialogue system.

1 Introduction

Detecting and adapting to user affect is being explored by many researchers to improve dialogue system quality. Detection has received much attention (e.g., (Litman and Forbes-Riley, 2004; Lee and Narayanan, 2005)), but less work has been done on adaptation, due to the difficulty of developing responses and applying them at the right time. Most work on adaptation takes a context-independent approach: use the same type of response after all instances of an affective state. For example, Liu and Picard (2005)'s health assessment system responds with empathy to all instances of user stress.

Research suggests, however, that it may be more effective to take a *context-dependent* approach: develop multiple responses for each affective state, whose use depends on the state's context. E.g., in the tutoring domain, Pon-Barry et al. (2006) show that *human* tutors use multiple responses to uncertain student answers, depending on the answer's correctness and prior context. In the information-seeking domain, it is commonly believed that while an apology is a good default response to user frustration (as in (Klein et al., 2002)), one context requires a different response: after several frustrated user turns, the call should be forwarded to a human operator.

A context-dependent approach to affect adaptation must address 2 issues: in what contexts to adapt, and what responses to use there. This paper addresses the first issue and targets student uncertainty in our computer tutoring dialogues. Although our dialogues have a Question-Answer format, our system contains 275 tutor questions. Treating each question as a context is too labor-intensive for adaptation development and creates a data sparsity issue. Instead we treat automatically monitorable question properties as contexts. Here we examine 3 contexts: the dialogue act interpretation, and the discourse structure depth and transition, of the prior tutor question. We use χ^2 to investigate the context dependency of uncertain student answers (correct and incorrect). Our results show that some contexts are significantly associated with uncertain answers. Our next step will be to use these significant dependencies to develop system responses to uncertain answers in these contexts. These responses will be based both on our hypotheses about why uncertainty is associated with these contexts, and on analyses of human tutor responses to uncertain answers in these contexts.

2 Student Uncertainty and Prior Context

ITSPOKE is a *speech-enabled* version of a text-based tutoring system (VanLehn et al., 2002). The student first types an essay answering one of 5 qualitative physics problems. ITSPOKE parses the essay, extracts dialogue topics concerning misconceptions, then engages the student in dialogue. In this study we used 2 ITSPOKE corpora containing 4590 student turns over 185 dialogues from 37 students. Figure 1 shows an annotated dialogue excerpt.

Proceedings of NAACL HLT 2007, Companion Volume, pages 41–44,
Rochester, NY, April 2007.

ITSPOKE$_4$: Good. Does the packet have an acceleration? If yes please specify its direction. *[SAQ, Advance, D1]*

STUDENT$_4$: horizontally forward yes *[non-uncertain, incorrect]*

ITSPOKE$_5$: Well... To find the packet's acceleration we will use Newton's 2nd law. What is the definition of this law? *[LAQ, Push, D2]*

STUDENT$_5$: an object in motion tends to stay in motion until it's act by an outside force *[uncertain, incorrect]*

ITSPOKE$_6$: Well... Newton's 2nd law says that the net force acting on an object equals the mass times the acceleration of the object. We can define Newton's 2nd law as F=m*a. Hence what can we determine about the acceleration of the packet given that we know the net force acting on it is not zero and points straight down? *[DAQ, Advance, D2]*

STUDENT$_6$: I don't know *[uncertain, incorrect]*

ITSPOKE$_7$: Well... < *further discussion of Newton's 2nd law...*> So once again on the original question. Does the packet have an acceleration? If yes please specify its direction.*[SAQ, PopUp, D1]*

Figure 1: ITSPOKE Annotated Dialogue Excerpt

2.1 Uncertainty and Correctness Annotations

ITSPOKE, like most computer tutors, responds only to student correctness. ITSPOKE labels each answer as *correct* or *incorrect*[1]. If correct, ITSPOKE moves on to the next question. If incorrect, then for questions on simple topics, ITSPOKE gives the correct answer and moves on, while for questions on complex topics (ITSPOKE$_4$, Figure 1), ITSPOKE initiates a sub-dialogue with remediation questions (ITSPOKE$_5$ - ITSPOKE$_6$), before moving on.

Recent computer tutoring research has shown interest in responding to student affect[2] over correctness. Uncertainty is of particular interest: researchers hypothesize that uncertainty and incorrectness each create an opportunity to learn (VanLehn et al., 2003). They cannot be equated, however. First, an uncertain answer may be correct or incorrect (Pon-Barry et al., 2006). Second, uncertainty indicates that the student *perceives* a possible misconception in their knowledge. Thus, system responses to uncertain answers can address both the correctness and the perceived misconception.

In our ITSPOKE corpora, each student answer has been manually annotated as *uncertain* or *non-uncertain*[3]: *uncertain* is used to label answers expressing uncertainty or confusion about the material; *non-uncertain* is used to label all other answers.

[1]We have also manually labeled correctness in our data; agreement between ITSPOKE and human is 0.79 Kappa (90%).

[2]We use 'affect' to cover emotions and attitudes that affect how students communicate. Although some argue 'emotion' and 'attitude' should be distinguished, some speech researchers find the narrow sense of 'emotion' too restrictive because it excludes states where emotion is present but not full-blown, including arousal and attitude (Cowie and Cornelius, 2003).

[3]A second annotator relabeled our dataset, yielding inter-annotator agreement of 0.73 Kappa (92%).

2.2 Context Annotations

Here we examine 3 automatically monitorable tutor question properties as our contexts for uncertainty:

Tutor Question Acts: In prior work one annotator labeled 4 Tutor Question Acts in one ITSPOKE corpus (Litman and Forbes-Riley, 2006)[4]: *Short (SAQ), Long (LAQ), and Deep Answer Question (DAQ)* distinguish the question in terms of content and the type of answer it requires. *Repeat (RPT)* labels variants of "Can you repeat that?" after rejections. From these annotations we built a hash table associating each ITSPOKE question with a Question Act label; with this table we automatically labeled ITSPOKE questions in our second ITSPOKE corpus.

Discourse Structure Depth/Transition: In prior work we showed that the discourse structure Depth and Transition for each ITSPOKE turn can be automatically annotated (Rotaru and Litman, 2006). E.g., as shown in Figure 1, ITSPOKE$_{4,7}$ have depth 1 and ITSPOKE$_{5,6}$ have depth 2. We combine levels 3 and above (3+) due to data sparsity. 6 Transition labels represent the turn's position relative to the prior ITSPOKE turn: *NewTopLevel* labels the first question after an essay. *Advance* labels questions at the same depth as the prior question (ITSPOKE$_{4,6}$). *Push* labels the first question in a sub-dialogue (after an incorrect answer) (ITSPOKE$_5$). After a sub-dialogue, ITSPOKE asks the original question again, labeled *PopUp* (ITSPOKE$_7$), or moves on to the next question, labeled *PopUpAdv*. *SameGoal* labels both ITSPOKE RPTS (after rejections) and repeated questions after timeouts.

[4]Our Acts are based on related work (Graesser et al., 1995). Two annotators labeled the Acts in 8 dialogues in a parallel human tutoring corpus, with agreement of 0.75 Kappa (90%).

3 Uncertainty Context Dependencies

We use the χ^2 test to investigate the context dependency of uncertain (unc) or non-uncertain (nonunc) student answers that are correct (C) or incorrect (I). First, we compute an overall χ^2 value between each context variable and the student answer variable. For example, the Question Act variable (QACT) has 4 values: *SAQ, LAQ, DAQ, RPT*. The answer variable (SANSWER) also has 4 values: *uncC, uncI, nonuncC, nonuncI*. Table 1 (last column) shows the χ^2 value between these variables is 203.38, which greatly exceeds the critical value of 16.92 ($p \leq 0.05$, df=9), indicating a highly significant dependency. Significance increases as the χ^2 value increases.

Dependency		Obs.	Exp.	χ^2
QACT ∼ SANSWER				203.38
LAQ ∼ uncC	+	72	22	133.98
LAQ ∼ uncI	+	43	27	11.17
LAQ ∼ nonuncC	-	96	151	50.13
LAQ ∼ nonuncI	=	48	60	3.10
DAQ ∼ uncC	=	22	22	0.01
DAQ ∼ uncI	+	37	27	4.57
DAQ ∼ nonuncC	=	135	149	3.53
DAQ ∼ nonuncI	=	63	59	0.35
SAQ ∼ uncC	-	285	328	41.95
SAQ ∼ uncI	-	377	408	17.10
SAQ ∼ nonuncC	+	2368	2271	66.77
SAQ ∼ nonuncI	-	875	898	5.31
RPT ∼ uncC	-	7	14	4.15
RPT ∼ uncI	=	22	18	1.25
RPT ∼ nonuncC	-	70	98	20.18
RPT ∼ nonuncI	+	70	39	33.59

Table 1: Tutor Question Act Dependencies ($p \leq .05$: critical χ^2=16.92 (df=9); critical χ^2=3.84 (df=1))

However, this does not tell us which variable values are significantly dependent. To do this, we create a binary variable from each value of the context and answer variables. E.g., the binary variable for *LAQ* has 2 values: "LAQ" and "Anything Else", and the binary variable for *uncC* has 2 values: "uncC" and "Anything Else". We then compute the χ^2 value between the binary variables. Table 1 shows this value is 133.98, which greatly exceeds the critical value of 3.84 ($p \leq 0.05$, df=1). The table also shows the observed (72) and expected (22) counts. Comparison determines the sign of the dependency: *uncC* occurs significantly more than expected (+) after LAQ. The "=" sign indicates a non-significant dependency.

Table 1 shows uncertain answers (*uncC* and *uncI*) occur significantly more than expected after LAQs. In contrast, non-uncertain answers occur significantly less (-), or aren't significantly dependent (=). Also, *uncI* occurs significantly more than expected after DAQs. We hypothesize that LAQs and DAQs are associated with more uncertainty because they are harder questions requiring definitions or deep reasoning. Not surprisingly, uncertain (and incorrect) answers occur significantly less than expected after SAQs (easier fill-in-the-blank questions). Uncertainty shows very weak dependencies on RPTs.

Table 2 shows that Depth1 is associated with more correctness and less uncertainty overall. Both types of correct answer occur significantly more than expected, but this dependency is stronger for *nonuncC*. Both incorrect answers occur significantly less than expected, but this dependency is stronger for *uncI*.

Dependency		Obs.	Exp.	χ^2
Depth# ∼ SANSWER				53.85
Depth1 ∼ uncC	+	250	228	5.46
Depth1 ∼ uncI	-	230	283	27.55
Depth1 ∼ nonuncC	+	1661	1579	24.73
Depth1 ∼ nonuncI	-	575	625	12.66
Depth2 ∼ uncC	-	78	101	7.80
Depth2 ∼ uncI	+	156	125	11.26
Depth2 ∼ nonuncC	-	664	699	5.65
Depth2 ∼ nonuncI	+	304	277	4.80
Depth3+ ∼ uncC	=	58	57	0.05
Depth3+ ∼ uncI	+	93	70	9.76
Depth3+ ∼ nonuncC	-	344	391	15.66
Depth3+ ∼ nonuncI	+	177	155	4.94

Table 2: Depth Dependencies ($p \leq .05$: critical χ^2=12.59 (df=6); critical χ^2=3.84 (df=1))

At Depths 2 and 3+, correct answers occur significantly less than expected or show no significance. Incorrect answers occur significantly more than expected, and the dependencies are stronger for *uncI*. We hypothesize that deeper depths are associated with increased uncertainty and incorrectness because they correspond to deeper knowledge gaps; uncertainty here may also relate to a perceived lack of cohesion between sub-topic and larger solution.

Table 3 shows Pushes have the same dependencies as deeper depths (increased uncertainty and incorrectness); however, here the *uncI* dependency is only slightly stronger than *nonuncI*, which suggests that increased uncertainty at deeper depths is more reliably associated with remediation questions after the Push. Although uncertainty shows only weak

dependencies on PopUps, after PopUpAdvs the *uncI* dependency is strong, with *uncI* occurring more than expected. We hypothesize that this dependency relates to students losing track of the original question/larger topic. Uncertainty shows only weak dependencies on Advances. After NewTopLevels, incorrect answers occur less than expected, but the dependency is stronger for *nonuncI*. After SameGoals, incorrect answers occur more than expected, but the dependency is stronger for *nonuncI*. Compared with the RPT results, the SameGoal results suggest students feel increased uncertainty after timeouts.

Dependency		Obs.	Exp.	χ^2
TRANS $\sim$ SANSWER				190.97
Push $\sim$ uncC	=	68	57	2.89
Push $\sim$ uncI	+	100	70	16.37
Push $\sim$ nonuncC	-	313	392	44.51
Push $\sim$ nonuncI	+	193	155	14.13
PopUp $\sim$ uncC	-	23	36	5.89
PopUp $\sim$ uncI	-	32	45	4.68
PopUp $\sim$ nonuncC	=	260	251	0.81
PopUp $\sim$ nonuncI	+	117	99	4.47
PopUpAdv $\sim$ uncC	=	8	13	2.50
PopUpAdv $\sim$ uncI	+	32	17	16.22
PopUpAdv $\sim$ nonuncC	-	76	93	7.72
PopUpAdv $\sim$ nonuncI	=	44	37	1.89
Advance $\sim$ uncC	=	217	205	1.70
Advance $\sim$ uncI	-	223	254	9.06
Advance $\sim$ nonuncC	+	1465	1416	8.66
Advance $\sim$ nonuncI	-	530	560	4.51
NewTopLevel $\sim$ uncC	=	53	54	0.04
NewTopLevel $\sim$ uncI	-	49	67	6.47
NewTopLevel $\sim$ nonuncC	+	463	375	57.33
NewTopLevel $\sim$ nonuncI	-	80	148	47.63
SameGoal $\sim$ uncC	=	17	21	0.70
SameGoal $\sim$ uncI	+	43	25	14.24
SameGoal $\sim$ nonuncC	-	92	152	44.25
SameGoal $\sim$ nonuncI	+	92	56	31.43

Table 3: Transition Dependencies ($p \leq .05$: critical χ^2=25.00 (df=15); critical χ^2=3.84 (df=1))

4 Current Directions

We analyzed dependencies between uncertain student answers and 3 automatically monitorable contexts. We plan to examine more contexts, such as a Topic Repetition variable that tracks similar questions about a topic (e.g. gravity) across dialogues.

Our next step will be to use the significant dependencies to develop system responses to uncertain answers in these contexts. These responses will be based both on our hypotheses about why uncertainty is significantly associated with these contexts, as well as on analyses of human tutor responses in these contexts, using our human tutoring corpus, which was collected with our first ITSPOKE corpus using the same experimental procedure.

We also plan to investigate context dependencies for other affective states, such as student frustration.

Acknowledgments

NSF (#0631930, #0354420 and #0328431) and ONR (N00014-04-1-0108) support this research.

References

R. Cowie and R. R. Cornelius. 2003. Describing the emotional states that are expressed in speech. *Speech Communication*, 40:5–32.

A. Graesser, N. Person, and J. Magliano. 1995. Collaborative dialog patterns in naturalistic one-on-one tutoring. *Applied Cognitive Psychology*, 9:495–522.

J. Klein, Y. Moon, and R. Picard. 2002. This computer responds to user frustration: Theory, design, and results. *Interacting with Computers*, 14:119–140.

C. M. Lee and S. Narayanan. 2005. Towards detecting emotions in spoken dialogs. *IEEE Transactions on Speech and Audio Processing*, 13(2), March.

D. Litman and K. Forbes-Riley. 2004. Predicting student emotions in computer-human tutoring dialogues. In *Proc. ACL*, pages 352–359.

D. J. Litman and K. Forbes-Riley. 2006. Correlations between dialogue acts and learning in spoken tutoring dialogues. *Natural Language Engineering*, 12(2).

K. Liu and R. W. Picard. 2005. Embedded empathy in continuous, interactive health assessment. In *CHI Workshop on HCI Challenges in Health Assessment*.

H. Pon-Barry, K. Schultz, E. Bratt, B. Clark, and S. Peters. 2006. Responding to student uncertainty in spoken tutorial dialogue systems. *International Journal of Artificial Intelligence in Education*, 16:171–194.

M. Rotaru and D. Litman. 2006. Exploiting discourse structure for spoken dialogue performance analysis. In *Proceedings of EMNLP*, Sydney, Australia.

K. VanLehn, P. W. Jordan, and C. P. Rosé et al. 2002. The architecture of Why2-Atlas: A coach for qualitative physics essay writing. In *Proceedings of ITS*.

K. VanLehn, S. Siler, and C. Murray. 2003. Why do only some events cause learning during human tutoring? *Cognition and Instruction*, 21(3):209–249.

A Filter-Based Approach to Detect End-of-Utterances from Prosody in Dialog Systems

Olac Fuentes, David Vera and Thamar Solorio
Computer Science Department
University of Texas at El Paso
El Paso, TX 79968

Abstract

We propose an efficient method to detect end-of-utterances from prosodic information in conversational speech. Our method is based on the application of a large set of binary and ramp filters to the energy and fundamental frequency signals obtained from the speech signal. These filter responses, which can be computed very efficiently, are used as input to a learning algorithm that generates the final detector. Preliminary experiments using data obtained from conversations show that an accurate classifier can be trained efficiently and that good results can be obtained without requiring a speech recognition system.

1 Introduction

While there have been improvements and a significant number of methods introduced into the realm of dialog-based systems, there are aspects of these methods which can be further improved upon. One such aspect is end-of-utterance (EOU) detection, which consists of automatically determining when a user has finished his/her turn and is waiting to receive an answer from the system. Current dialog-based systems use a simple pause threshold, which commonly results in either unnecessary long waiting times or interruptions from the system when the user makes a pause in the middle of an utterance. These problems can annoy and discourage users using even simple dialog systems.

Most previous methods aimed at improving upon pause thresholds for detecting end-of-utterances use spectral energy measures (Hariharan et al., 2001; Jia and Xu, 2002). Other methods use prosodic features with (Ferrer et al., 2002) and without speech recognition systems (Ferrer et al., 2003) in conjunction with decision trees to determine end-of-utterances as quickly as possible. For this and related problems, the choice of features is critical. Most common is to use a fixed inventory of features, chosen based on the linguistics literature and past experience (Shriberg and Stolcke, 2004). Recently we have experimented with alternative approaches, including features hand-tailored to specific discrimination problems (Ward and Al Bayyari, 2006) and random exploration of the feature space (Solorio et al., 2006). In this paper we explore yet another approach, using a large battery of very simple and easy to evaluate features.

In this paper we present a method to improve the accuracy that can be obtained in end-of-utterance detection that uses prosodic information only, without a speech recognizer. We adapt and extend a filter-based approach originally proposed in computer graphics (Crow, 1984) and later exploited successfully in computer vision (Viola and Jones, 2001) and music retrieval (Ke et al., 2005).

Our approach consists of applying simple filters, which can be computed in constant time, in order to generate attributes to be used by a learning algorithm. After the attributes have been generated, we test different learning algorithms to detect end-of-utterances. Our results show that the features yield good results in combination with several of the classifiers, with the best result being obtained with bagging ensembles of decision trees.

2 Method

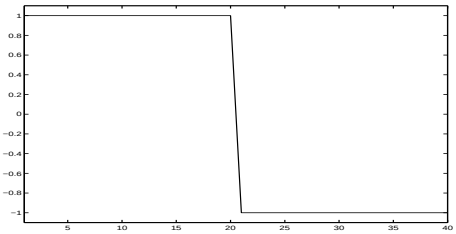
The first stage in our system is to extract prosodic information from the raw audio signal. Using the audio analysis tool Didi, the log energy and fundamental frequency signals are extracted from the source sound wave. After computing log energy and pitch, we apply a large set of filters in the time domain to the energy and pitch signals in order to generate attributes suitable for classification. We compute the filter responses for both signals at every time step using three types of filters, each applied at many different times scales.

The first filter type, shown in Figure 1a), is a two-step binary filter, split approximately in half. The first half of the filter consists of a sequence of 1's. The second half consists of -1's. The second filter type is a three-step binary filter (as shown in Figure 1b)), split in approximate thirds alternating between 1 and -1. Finally, the third filter is an upward slope ranging from -1 to 1.

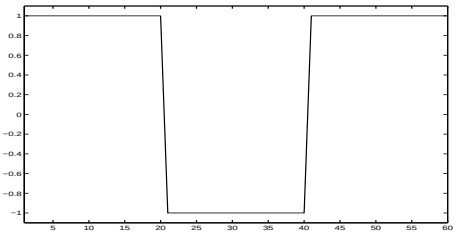
Although simple, these filters, in particular when they are applied at multiple scales, can characterize most of the prosodic features that are known to be relevant in identifying dialog phenomena including raises and falls in pitch and pauses of different lengths.

The response of any of these filters over the signal at any time is given by the dot product of the filter and signal

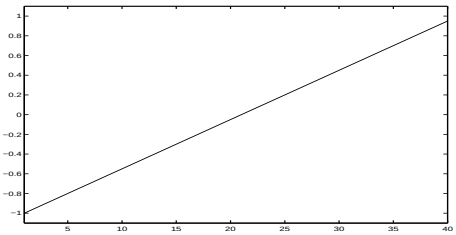
Proceedings of NAACL HLT 2007, Companion Volume, pages 45–48,
Rochester, NY, April 2007.



a) Type I filter



b) Type II filter



c) Type III filter

Figure 1: The three types of filters, the first two being binary only having values -1 or 1, and the last having an upward slope from -1 to 1.

window of the same length. Computing this dot product is slow, especially over larger time window sizes. This cost is even greater when many filter responses are taken over the course of the entire signal length.

Given the large number of filters and the size of a normal audio signal, the straightforward dot-product-based computation of the filter responses is prohibitively expensive. Fortunately, it is possible to device methods to compute these responses efficiently, as explained in the next subsection.

2.1 Efficient Filter Computation

This constant time computation of binary filters for two-dimensional signals was first presented by Crow (Crow, 1984) in the field of computer graphics and later applied successfully in computer vision (Viola and Jones, 2001). Here we show how that can be adapted to one-dimensional signals and extended to the case of non-binary filters, such as ramps.

Let s be the signal corresponding to either the log energy or the fundamental frequency. Let f be a filter of size n (in arbitrary time units) and let k be the time instant for which we want to compute the filter response.

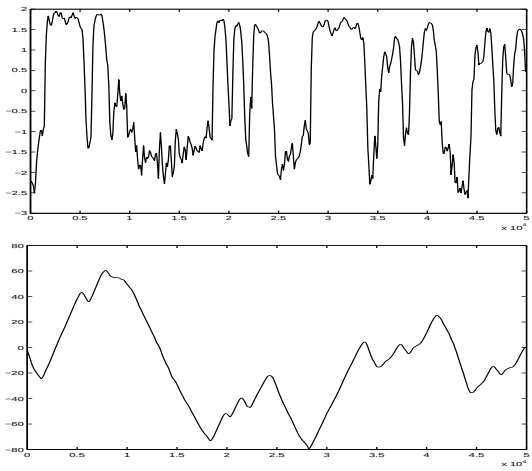


Figure 2: Energy, with mean subtracted, and its corresponding integral signal.

The filter response F is given by

$$F(s, f, k) = \sum_{i=0}^{n-1} s_{k+i} * f_i$$

The standard computation of F takes $O(n)$ operations; however, for the special case of binary filters like the ones shown in figures 1a) and 1b), we can compute this response in constant time with some preprocessing as follows. Let I be the integral signal, where each element of I is given by

$$I_j = \sum_{i=0}^{j} s_i$$

It can be seen that

$$\sum_{i=j}^{k} s_j = I_k - I_{j-1}$$

Thus this summation can be computed with two accesses to memory, after pre-computing and storing the values of I in an array. Figure 2 shows an example of a signal (with its mean subtracted) and the corresponding integral signal.

Consider a binary filter f such as the one shown in 1a), $f = \{1^{n/2}, -1^{n/2}\}$, that is, f consists of $n/2$ ones followed by $n/2$ negative ones. Then the filter response of a signal can then be computed in constant time using three references to the integral signal I:

$$F(s, f, k) = 2I_{k+n/2-1} - I_{k-1} - I_{k+n-1}$$

Similarly, the response to a filter like the one shown in Figure 1 b), given by $f = \{1^{n/3}, -1^{n/3}, 1^{n/3}\}$ can be computed with four memory references.

$$F(s, f, k) = I_{k+n-1} - 2I_{k+2n/3-1} + 2I_{k+n/3-1} - I_{k-1}$$

The third filter is an upward ramp ranging from -1 to 1. Whereas the binary filters are simple to calculate using look-up values, and their application to 1-dimensional signals is a simple adaptation to the 2-D algorithm, a ramp is more difficult and requires separate preprocessing for filters of different lengths. Regardless, it is still possible to compute its response in constant time after preprocessing.

We define a ramp filter of length n as $f = \{-1, \frac{2}{n-1} - 1, \frac{4}{n-1} - 1, ..., 1 - \frac{2}{n-1}, 1\}$. The response to this filter is

$$F(s, f, k) = \sum_{i=0}^{n-1} s_{k+i} * f_i$$

$$= \sum_{i=0}^{n-1} (\frac{2i}{n-1} - 1) s_{k+i}$$

$$= \frac{2}{n-1} \sum_{i=0}^{n-1} i s_{k+i} - \sum_{i=0}^{n-1} s_{k+i}$$

$$= \frac{2}{n-1} \sum_{i=0}^{n-1} i\, s_{k+i} - (I_{n-1} - I_{k-1})$$

Let $\sum_{i=0}^{n-1} i\, s_{k+i}$ be denoted by An_k. Clearly, if An_k can be computed in constant time, then $F(s, f, k)$ can also be computed in constant time. This can be done, with a little preprocessing, as follows. Let An_0 be computed in the standard ($O(n)$ time) way,

$$An_0 = \sum_{i=0}^{n-1} i s_i$$

Then to compute values of An_k for $k > 0$ we first observe that

$$An_k = s_{k+1} + 2s_{k+2} + \ldots + (n-1)s_{k+n-1}$$

and

$$An_{k+1} = s_{k+2} + 2s_{k+3} + \ldots + (n-1)s_{k+n}$$

From this, we can see that

$$An_{k+1} = An_k - s_{k+1} - s_{k+2} - \ldots$$

$$-s_{k+n-1} + (n-1)s_{k+n}$$

$$= An_k - \sum_{i=k+1}^{k+n-1} s_i + (n-1)s_{k+n}$$

$$= An_k - (I_{k+n-1} - I_k) + (n-1)s_{k+n}$$

Thus after pre-computing vectors I and An, which takes linear time in the size of the signal, we can compute any filter response in constant time. However, while we can derive all binary-filter responses from vector I, computing ramp-filter responses requires the pre-computation of a separate An for every filter length n. Nevertheless, this cost is small compared to the cost of computing a dot product for every time instant in the input signal.

The integral signal representations are computed from the two prosodic feature signals, and filter features are calculated along the timeframe of the signal. Once the filter responses are obtained, they are used as attributes for machine learning algorithms, which are used to generate the final classifiers. The data is then used to train several learning algorithms, as implemented in Weka (Witten and Frank, 2005).

3 Experimental Results

Experiments were conducted to test the end-of-utterance system on pre-recorded conversations, measuring precision and recall of positive classifications. The conversations used contained speech by both male and female users to compare the robustness among different vocal range frequencies.

3.1 Data

The training set of data is derived from about 22 minutes of conversations, with the audio split such that each speaker is on a separate voice channel (see (Hollingsed, 2006)). In 17-minutes worth of these, volunteers were asked to list eight exits on the east side of El Paso on Interstate 10. This provided a clear way to measure end-of-utterances as if a system were prompting users for input. This set of conversations contained a large number of turn-switches, which also simulated voice-portal systems well. For most of the time in this set, the same person (a female) is conducting the quiz. However, the speakers taking the quiz have distinctly different voices and are mixed in gender.

Five minutes of the training set were taken from a casual conversation also containing a male and female speaker combination. The speakers in this conversation are different from the speakers in the other dataset. Adding these data balances the training set, reducing the probability of the system learning only the specific quiz format used in much of the training data.

Didi was used to extract the prosodic features, and the filter responses were computed for each of the three filter types, in sizes ranging from 200ms to 3 seconds in increments of 50ms, totaling 342 features per time instance. The class was set to 0 or 1, signaling non-end-of-utterances and a confirmed end-of-utterances, respectively. 992 instances were created for the experiments, split equally in two between positive examples of end-of-utterances, and randomly selected negative examples for both channels in the source audio.

Table 1: Experimental results of using different classifiers and averaging ten ten-fold-cross-validation evaluations with random seeds per classifier.

	Recall	Precision	F-Measure
Dec.Stump	0.623	0.705	0.660
Dec.Table	0.768	0.799	0.783
C4.5	0.792	0.800	0.796
Boost(DS)	0.792	0.820	0.806
Bag(REP)	0.850	0.833	0.841
Bag(C4.5)	0.786	0.797	0.791

All instances used for training were randomly chosen. The positive examples were chosen from human-determined end-of-utterance intervals, which ranged from the time instant a valid end-of-utterance was recorded to a point either 1.5 seconds after that instant or a start-of-utterance that occurred prior to that time. The negative examples were randomly chosen such that no time instance was chosen prior to the 3-second-mark of the audio file used and none was within a marked end-of-utterance interval.

3.2 Results

Six combinations of classifiers were generated using the Weka data mining tool. Each of these classifier combinations was tested using 10-fold cross-validation. The results reflect the average of ten such cross-validation runs, each using a different random seed. The final classifier combinations used are Weka's implementations of decision stumps, decision tables, C4.5 (Quinlan, 1993) and ensembles of decision stumps using boosting and C4.5 and reduced error pruning (REP) decision trees (Quinlan, 1987) using bagging.

The experiments performed yield interesting results. Table 1 shows that, with the exception of decision stumps, which are perhaps too simple for this task, all classifiers performed well, which shows that our filters produce suitable features for classification. The best results were obtained using bagging and REP trees, but results for other methods yield similar precision and recall.

It is almost certain that better results can be obtained using these methods if bleeding across channels in the audio streams was reduced. The F0 features do a good job of filtering out possible mistakes in the system due to the way the frequencies are calculated. However, bleeding can still mislead the classifiers into perceiving an end-of-utterance from another speaker.

4 Conclusions and Future Work

We have shown a new filter-based method for detecting end-of-utterances in conversation using only basic prosodic information. We adapted and extended previously described methods for fast computation of filter responses, which allows our system to be trained quickly and easily permits real-time performance. Preliminary experiments in the task of classifying windows in dialog recordings as being end-of-utterances or not have yielded very promising results using standard classification algorithms, with an f-measure of 0.84.

Present and future work includes evaluating the method as a component of a real-time dialog system, where its usefulness at decreasing waiting time can be tested. We are also working on methods for feature selection and compression to obtain further speedup, and finally we are experimenting with larger datasets.

Acknowledgement: The authors would like to thank NSF for partially supporting this work under grants IIS-0415150 and 0080940.

References

F. C. Crow. 1984. Summed-area tables for texture mapping. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*.

L. Ferrer, E. Shriberg, and A. Stolcke. 2002. Is the speaker done yet? Faster and more accurate end-of-utterance detection using prosody. In *Proceedings of ICSLP*.

L. Ferrer, E. Shriberg, and A. Stolcke. 2003. A prosody-based approach to end-of-utterance detection that does not require speech recognition. In *Proceedings of IEEE ICASSP*.

R. Hariharan, J. Hakkinen, and K. Laurila. 2001. Robust end-of-utterance detection for real-time speech recognition applications. In *Proceedings of IEEE ICASSP*.

T. K. Hollingsed. 2006. Responsive behavior in tutorial spoken dialogues. Master's thesis, University of Texas at El Paso.

C. Jia and B. Xu. 2002. An improved entropy-based endpoint detection algorithm. In *Proceedings of ISCSLP*.

Y. Ke, D. Hoiem, and R. Sukthankar. 2005. Computer vision for music identification. In *Proceedings of IEEE CVPR*.

J. R. Quinlan. 1987. Simplifying decision trees. *International Journal of Man-Machine Studies*, 27.

J. R. Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufman.

E. Shriberg and A. Stolcke. 2004. Direct modeling of prosody: An overview of applications in automatic speech processing. In *Proceedings of ICSP*.

T. Solorio, O. Fuentes, N. Ward, and Y. Al Bayyari. 2006. Prosodic feature generation for back-channel prediction. In *Proceedings of Interspeech*.

P. Viola and M. Jones. 2001. Rapid object detection using a boosted cascade of simple features. In *Proceedings of IEEE CVPR*.

N. Ward and Y. Al Bayyari. 2006. A case study in the identification of prosodic cues to turn-taking: Back-channeling in Arabic. In *Proceedings of Interspeech*.

I. H. Witten and E. Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufman.

Document Similarity Measures to Distinguish Native vs. Non-Native Essay Writers

Olga Gurevich
Educational Testing Service
Rosedale & Carter Roads,
Turnbull 11R
Princeton, NJ 08541
ogurevich@ets.org

Paul Deane
Educational Testing Service
Rosedale & Carter Roads,
Turnbull 11R
Princeton, NJ 08541
pdeane@ets.org

Abstract

The ability to distinguish statistically different populations of speakers or writers can be an important asset in many NLP applications. In this paper, we describe a method of using document similarity measures to describe differences in behavior between native and non-native speakers of English in a writing task.[1]

1 Introduction

The ability to distinguish statistically different populations of speakers or writers can be an important asset in many NLP applications. In this paper, we describe a method of using document similarity measures to describe differences in behavior between native and non-native speakers of English in a prompt response task.

We analyzed results from the new TOEFL integrated writing task, described in the next section. All task participants received the same set of prompts and were asked to summarize them. The resulting essays are all trying to express the same "gist" content, so that any measurable differences between them must be due to differences in individual language ability and style. Thus the task is uniquely suited to measuring differences in linguistic behavior between populations.

Our measure of document similarity, described in section 3, is a combination of word overlap and syntactic similarity, also serving as a measure of syntactic variability. The results demonstrate significant differences between native and non-native speakers that cannot be attributed to any demographic factor other than the language ability itself.

2 TOEFL Integrated Writing Task and Scoring

The Test of English as a Foreign Language (TOEFL) is administered to foreign students wishing to enroll in US or Canadian universities. It aims to measure the extent to which a student has acquired English; thus native speakers should on average perform better on the test regardless of their analytical abilities. The TOEFL now includes a writing component, and pilot studies were conducted with native as well as non-native speakers.

One of the writing components is an Integrated Writing Task. Students first read an expository passage, which remains on the screen throughout the task. Students then hear a segment of a lecture concerning the same topic. However, the lecture contradicts and complements the information contained in the reading. The lecture is heard once; students then summarize the lecture and the reading and describe any contradictions between them.

The resulting essays are scored by human raters on a scale of 0 to 5, with 5 being the best possible score[2]. The highest-scoring essays express ideas from both the lecture and the reading using correct grammar; the lowest-scoring essays rely on only one of the prompts for information and have grammatical problems; and the scores in between show a combination of both types of deficiencies.

The test prompt contained passages about the advantages and disadvantages of working in groups; the reading was 260 words long, the lecture 326 words. 540 non-native speakers and 950

[1] This research was funded while the first author was a Research Postdoctoral Fellow at ETS in Princeton, NJ.

[2] Native speaker essays were initially scored with possible half-grades such as 2.5. For purposes of comparison, these were rounded down to the nearest integer.

Proceedings of NAACL HLT 2007, Companion Volume, pages 49–52,
Rochester, NY, April 2007.

native speakers were tested by ETS in 2004. ETS also collected essential demographic data such as native language, educational level, etc., for each student. For later validation, we excluded 1/3 of each set, selected at random, thus involving 363 non-native speakers and 600 native speakers.

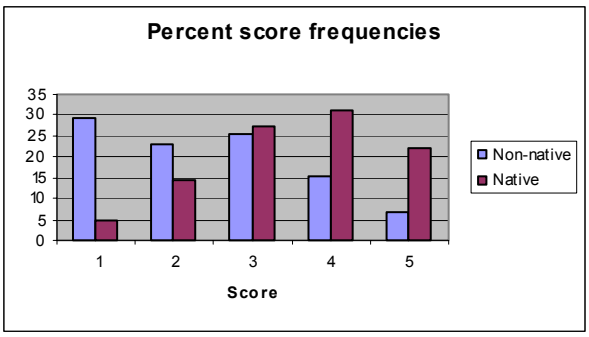


Figure 1. Relative score distributions.

Among the non-native speakers, the most common score was 1 (see Fig. 1 for a histogram). By contrast, native speaker scores centered around 3 and showed a normal-type distribution. The difference in distributions confirms that the task is effective at separating non-native speakers by skill level, and is easier for native speakers. The potential sources of difficulty include comprehension of the reading passage, listening ability and memory for the lecture, and the analytical ability to find commonalities and differences between the content of the reading and the lecture.

3 Document Similarity Measure

Due to the design of the TOEFL task, the content of the student essays is highly constrained. The aim of the computational measures is to extract grammatical and stylistic differences between different essays. We do this by comparing the essays to the reading and lecture prompts. Our end goal is to determine to what extent speakers diverge from the prompts while retaining the content.

The prediction is that native speakers are much more likely to paraphrase the prompts while keeping the same gist, whereas non-native speakers are likely to either repeat the prompts close to verbatim, or diverge from them in ways that do not preserve the gist. This intuition conforms to previous studies of native vs. non-native speakers' text summarization (cf. Campbell 1987), although we are not aware of any related computational work.

We begin by measuring lexico-grammatical similarity between each essay and the two prompts. An essay is represented as a set of features derived from its lexico-grammatical content, as described below. The resulting comparison measure goes beyond simple word or n-gram overlap by providing a measure of structural similarity as well. In essence, our method measures to what extent the essay expresses the content of the prompt in the same words, used in the same syntactic positions.

3.1 C-rater tuples

In order to get a measure of syntactic similarity, we relied on C-rater (Leacock & Chodorow 2003), an automatic scoring engine developed at ETS. C-rater includes several basic NLP components, including POS tagging, morphological processing, anaphora resolution, and shallow parsing. The parsing produces *tuples* for each clause, which describe each verb and its syntactic arguments (1).

(1) CLAUSE: the group spreads responsibility for a decision to all the members
TUPLE: *:verb:* spread *:subj:* the group *:obj:* responsible *:pp.for:* for a decide *:pp.to:* to all

C-rater does not produce full-sentence trees or prepositional phrase attachment. However, the tuples are reasonably accurate on non-native input.

3.2 Lexical and Syntactic Features

C-rater produces tuples for each document, often several per sentence. For the current experiment, we used the main verb, its subject and object. We then converted each tuple into a set of features, which included the following:

- The verb, subject (pro)noun, and object (pro)noun as individual words;
- All of the words together as a single feature;
- The verb, subject, and object words with their argument roles.

Each document can now be represented as a set of tuple-derived features, or feature vectors.

3.3 Document Comparison

Two feature vectors derived from tuples can be compared using a cosine measure (Salton 1989). The closer to 1 the cosine, the more similar the two feature sets. To compensate for different frequencies of the features and for varying document lengths, the feature vectors are weighted using standard *tf*idf* techniques.

In order to estimate the similarity between two documents, we use the following procedure. For each tuple vector in Document A, we find the tuple in Document B with the maximum cosine to the tuple in Document A. The maximum cosine values for each tuple are then averaged, resulting in a single scalar value for Document A. We call this measure *Average Maximum Cosine* (AMC).

We calculated AMCs for each student response versus the reading, the lecture, and the reading + lecture combined. This procedure was performed for both native and non-native essays. A detailed examination of the resulting trends is in section 4.

3.4 Other Measures of Document Similarity

We also performed several measures of document similarity that did not include syntactic features.

Content Vector Analysis

The student essays and the prompts were compared using Content Vector Analysis (CVA), where each document was represented as a vector consisting of the words in it (Salton 1989). The *tf*idf*-weighted vectors were compared by a cosine measure.

For non-native speakers, there was a noticeable trend. At higher score levels (where the score is determined by a human rater), student essays showed more similarity to both the reading and the lecture prompts. Both the reading and lecture similarity trends were significant (linear trend; $F=MS_{\text{linear trend}}/MS_{\text{within-subjects}}=63$ for the reading; F=71 for the lecture at 0.05 significance level[3]). Thus, the rate of vocabulary retention from both prompts increases with higher English-language skill level.

Native speakers showed a similar pattern of increasing cosine similarity between the essay and the reading (F=35 at 0.05 significance for the trend), and the lecture (F=35 at the 0.05 level).

BLEU score

In order to measure the extent to which whole chunks of text from the prompt are reproduced in the student essays, we used the BLEU score, known from studies of machine translation (Papineni et al. 2002). We used whole essays as sections of text rather than individual sentences.

For non-native speakers, the trend was similar to that found with CVA: at higher score levels, the overlap between the essays and both prompts increased (F=52.4 at the 0.05 level for the reading; F=53.6 for the lecture).

Native speakers again showed a similar pattern, with a significant trend towards more similarity to the reading (F=35.6) and the lecture (F=31.3). These results were confirmed by a simple n-gram overlap measure.

4 Results

4.1 Overall similarity to reading and lecture

The AMC similarity measure, which relies on syntactic as well as lexical similarity, produced somewhat different results from simpler bag-of-word or n-gram measures. In particular, there was a difference in behavior between native and non-native speakers: non-native speakers showed increased structural similarity to the lecture with increasing scores, but native speakers did not.

For non-native speakers, the trend of increased AMC between the essay and the lecture was significant (F=10.9). On the other hand, there was no significant increase in AMC between non-native essays and the reading (F=3.4). Overall, for non-native speakers the mean AMC was higher for the reading than for the lecture (0.114 vs. 0.08).

Native speakers, by contrast, showed no significant trends for either the reading or the lecture. Overall, the average AMCs for the reading and the lecture were comparable (0.08 vs. 0.075).

We know from results of CVA and BLEU analyses that for both groups of speakers, higher-scoring essays are more lexically similar to the prompts. Thus, the lack of a trend for native speakers must be due to lack of increase in structural similarity between higher-scoring essays and the prompts. Since better essays are presumably better at expressing the content of the prompts, we can hypothesize that native speakers paraphrase the content more than non-native speakers.

4.2 Difference between lecture and reading

The most informative measure of speaker behavior was the difference between the Average Maximum Cosine with the reading and the lecture, calculated by subtracting the lecture AMC from the reading AMC. Here, non-native speakers showed a significant downward linear trend with increasing

[3] All statistical calculations were performed as ANOVA-style trend analyses using SPSS.

score (F=6.5; partial eta-squared 0.08), whereas the native speakers did not show any trend (F=1.5). The AMC differences are plotted in Figure 3.

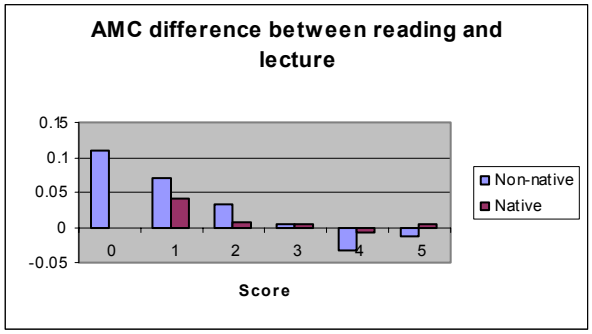


Figure 2 - AMC difference between reading and lecture

Non-native speakers with lower scores rely mostly on the reading to produce their response, whereas speakers with higher scores rely somewhat more on the lecture than on the reading. By contrast, native speakers show no correlation between score and reading vs. lecture similarity. Thus, there is a significant difference in the overall distribution and behavior between native and non-native speaker populations. This difference also shows that human raters rely on information other than simple verbatim similarity to the lecture in assigning the overall scores.

4.3 Other parameters of variation

For non-native speakers, the best predictor of the human-rated score is the difference in AMC between the reading and the lecture.

As demonstrated in the previous section, the AMC difference does not predict the score for native speakers. We analyzed native speaker demographic data in order to find any other possible predictors. The students' overall listening score, their status as monolingual vs. bilingual, their parents' educational levels all failed to predict the essay scores.

5 Discussion and Future Work

The Average Maximum Cosine measure as described in this paper successfully characterizes the behavior of native vs. non-native speaker populations on an integrated writing task. Less skillful non-native speakers show a significant trend of relying on the easier, more available prompt (the reading) than on the harder prompt (the lecture), whereas more skillful readers view the lecture as more relevant and rely on it more than on the reading. This difference can be due to better listening comprehension for the lecture and/or better memory. By contrast, native speakers rely on both the reading and the lecture about the same, and show no significant trend across skill levels. Native speakers seem to deviate more from the structure of the original prompts while keeping the same content, signaling better paraphrasing skills.

While not a direct measure of gist similarity, this technique represents a first step toward detecting paraphrases in written text. In the immediate future, we plan to extend the set of features to include non-verbatim similarity, such as synonyms and words derived by LSA-type comparison (Landauer et al. 1998). In addition, the syntactic features will be expanded to include frequent grammatical alternations such as active / passive.

A rather simple measure such as AMC has already revealed differences in population distributions for native vs. non-native speakers. Extensions of this method can potentially be used to determine if a given essay was written by a native or a non-native speaker. For instance, a statistical classifier can be trained to distinguish feature sets characteristic for different populations. Such a classifier can be useful in a number of NLP-related fields, including information extraction, search, and, of course, educational measurement.

References

Campbell, C. 1987. Writing with Others' Words: Native and Non-Native University Students' Use of Information from a Background Reading Text in Academic Compositions. Technical Report, UCLA Center for Language Education and Research.

Landauer, T.; Foltz, P. W; and Laham. D. 1998. Introduction to Latent Semantic Analysis. *Discourse Processes* 25: 259-284.

Leacock, C., & Chodorow, M. 2003. C-rater: Scoring of short-answer questions. *Computers and the Humanities,* 37(4), 389-405.

Papineni, K; Roukos, S.; Ward, T. and Zhu, W-J. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. *ACL* '02, p. 311-318.

Salton, G. 1989. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Reading, MA: Addison-Weley.

Arabic Diacritization through Full Morphological Tagging

Nizar Habash and Owen Rambow
Center for Computational Learning Systems
Columbia University
New York, NY 10115, USA
{habash,rambow}@cs.columbia.edu

Abstract

We present a diacritization system for written Arabic which is based on a lexical resource. It combines a tagger and a lexeme language model. It improves on the best results reported in the literature.

1 Introduction

Arabic is written without certain orthographic symbols, called **diacritics**, which represent among other things short vowels.[1] The restoration of diacritics to written Arabic is an important processing step for several natural language processing applications, including training language models for automatic speech recognition, text-to-speech generation, and so on. For a discussion of the role of diacritization, see (Maamouri et al., 2006). In this paper, we present a new diacritization module that outperforms the best previously published results, using a new combination of techniques. A more detailed presentation can be found in (Habash and Rambow 2007).

2 Diacritization in Arabic: Linguistic Description

Arabic script consists of two classes of symbols: letters and diacritics. Letters are always written whereas diacritics are optional: written Arabic can be fully diacritized, it can have some diacritics (to disambiguate certain words), or it can be entirely undiacritized. There are three types of diacritics: vowel, nunation, and shadda. Vowel diacritics represent Arabic's three short vowels and the absence of any vowel. The following are the four vowel-diacritics exemplified in conjunction with the letter ب *b* (we use Buckwalter transliteration): بَ *ba*, بُ *bu*, بِ *bi* and بْ *bo (no vowel)*. Nunation diacritics can only occur in word final positions in nominals (nouns, adjectives and adverbs). They represent a short vowel followed by an *n* sound: بًا[2] *bF*, بٌ *bN* and بٍ *bK*. Nunation is an indicator of nominal indefiniteness. Shadda is a consonant doubling diacritic: بّ *b∼*. The shadda can combine with vowel or nunation diacritics: بُّ *b∼u* or بٌّ *b∼N*. Additional *diacritical marks* in Arabic include the hamza, which appears in conjunction with a small number of letters (e.g., ئ , ؤ ,آ ,إ ,أ). Since most Arabic encodings do not consider the hamza a diacritic, but rather a part of the letter (like the dot on the lowercase Roman *i* or under the Arabic *b*: ب), we do not count it here as part of the diacritic set.

Functionally, diacritics can be split into two different kinds: **lexemic diacritics** and **inflectional diacritics**. Lexemic diacritics distinguish between two lexemes.[3] For example, the diacritization difference between the lexemes كَاتِب *kAtib* 'writer' and كَاتَب *kAtab* 'to correspond' distinguish between the meanings of the word rather than their inflections. Thus, there are lexemes that look alike when undiacritized but are spelled differently when diacritized. Note that there are also distinct lexemes that are always spelled the same way, even when diacritized – their difference is only a difference in word sense.

Inflectional diacritics distinguish different inflected forms of the *same* lexeme. For instance, the final diacritics in كَتَبْتُ *katabtu* 'I wrote' and كَتَبْتَ *katabta* 'you wrote' distinguish the person of the subject of the verb. We further distinguish be-

[1]This research was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023. Any opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of DARPA.

[2]Arabic orthography calls for adding a silent Alif (ا) in conjunction with ً in words ending with a consonant.

[3]A **lexeme** is an abstraction over inflected wordforms which groups together all those wordforms that differ only in terms of one of the morphological categories such as number, gender, aspect, or voice. The **lemma** is the distinguished word form which serves as citation form.

Proceedings of NAACL HLT 2007, Companion Volume, pages 53–56,
Rochester, NY, April 2007.

tween two types of inflectional diacritics: variant inflectional diacritics and invariant inflectional diacritics. The distinction is made with respect to two morphosyntactic features: nominal case and verbal mood. The variant inflectional diacritics need not always appear at the end of the word. For instance, the variant inflectional diacritics at the penultimate positions of the following two words distinguish their case: كَاتِبُهُ *kAtibuhu* 'his writer [nominative]' and كَاتِبَهُ *kAtibahu* 'his writer [accusative]'.

3 The MADA-D System

In a previous publication, we described the Morphological Analysis and Disambiguation of Arabic (MADA) system (Habash and Rambow, 2005). The basic approach used in MADA is inspired by the work of Hajič (2000) for tagging morphologically rich languages, which was extended to Arabic independently by Hajič et al. (2005). In this approach, a set of taggers are trained for individual linguistic features which are components of the full morphological tag (such as core part-of-speech, tense, number, and so on). In Arabic, we have ca. 2,000 to 20,000 morphological tags, depending on how we count. The Buckwalter Arabic Morphological Analyzer (BAMA) (Buckwalter, 2004) is consulted to produce a list of possible analyses for a word. BAMA returns, given an undiacritized inflected word form, all possible morphological analyses, *including full diacritization* for each analysis. The results of the individual taggers are used to choose among these possible analyses. The algorithm we proposed in (Habash and Rambow, 2005) for choosing the best BAMA analysis simply counts the number of predicted values for the set of linguistic features in each candidate analysis. Hajič et al. (2005), however, weigh the predicted values by their probability or confidence measure. To our knowledge, no results on diacritization have been previously reported using this particular approach to tagging.[4]

In this paper, we extend our basic MADA system in the following ways: First, we follow Hajič et al. (2005) in including case, mood, and nunation as features, because of its importance to diacritization. Second, we replace the YAMCHA (Kudo and Matsumoto, 2003) implementation of Support Vector Machines (SVMs) with SVMTool (Giménez and Màrquez, 2004) as our machine learning tool, for reasons of speed, at the cost of a slight decrease in accuracy. Like Hajič et al. (2005), we do not use Viterbi decoding. Finally, we introduce a specialized module for resolving residual ambiguity after the basic tagging is done. We explain this module in detail next. We train our classifiers on the exact training set defined by Zitouni et al. (2006), a subpart of the third segment of the Penn Arabic Treebank (Maamouri et al., 2004) ("ATB3-Train", 288,000 words). We also (reluctantly) follow them in having a single set for development and testing ("ATB3-Devtest", 52,000 words), rather than separate development and test sets (as is common), in order to be able to compare our results to theirs.

Up until this point, MADA-D has narrowed the list of possible analyses of a word (supplied by BAMA) down to a small number. This number can sometimes be greater than one for two reasons: first, the way in which we use the output of the taggers to choose among the analyses may yield a tie among several analyses; second, there may be lexeme-based diacritic ambiguity, and the morphological taggers cannot disambiguate lexemic diacritization. To address the residual ambiguity, we implemented a second component. Ideally, this would be (or include) a full word sense disambiguation (WSD) system, but WSD is a hard problem. Instead, we approximate WSD using standard n-gram language models. We use two types of data for training: fully diacritized word forms, and data in which we have replaced the inflected word by the diacritized citation form of the lexeme. Note that this procedure conflates lexemes that differ *only* in meaning, not in diacritization, as we are not actually interested in WSD for its own sake in this paper. The training corpus is the same corpus we use for the classifiers, ATB3-Train. This means that the diacritization and the choice of lexeme are done by hand, but it also means that the training set is quite small by the standards of language models. We build an open-vocabulary language model with Kneser-Ney smoothing using the SRILM toolkit (Stolcke, 2002). We will call the resulting language models X**LM-**n, where X is "D" for the fully diacritized word forms, or "L" for the lexeme citation forms, and n is the order of the n-

[4] Smith et al. (2005) also use the Buckwalter Analyzer in their Arabic morphological tagger, but then use a rather different approach to choosing among the possible analyses. They represent the possible analyses in a lattice, and a noisy channel model to choose among them. We leave to future work how the issue of diacritization can be integrated with their model.

grams ($n = 1, 2, 3$). When all candidate tokens (diacritized word or lexeme citation form) are unknown (out-of-vocabulary), the language model does not actually make a choice among them. We then use a diacritization unigram model, and then finally random choice. In the case of a preceding DLM-n model, this simply amounts to random choice, but in the case of a preceding LLM-n model, the diacritization model may actually make a non-random choice.

4 Related Work

We review three approaches that are directly relevant to us; we refer to the excellent literature review in (Zitouni et al., 2006) for a general review. Vergyri and Kirchhoff (2004) follow an approach similar to ours in that they choose from the diacritizations proposed by BAMA. However, they train a single tagger using unannotated data and EM, which necessarily leads to a lower performance. The most salient difference, however, is that they are motivated by the goal of improving automatic speech recognition, and have an acoustic signal parallel to the undiacritized text. All their experiments use acoustic models. They show that WER for diacritization decreases by nearly 50% (from 50%) when BAMA is added to the acoustic information, but the tagger does not help. It would be interesting to investigate ways of incorporating acoustic model information in our approach.

Ananthakrishnan et al. (2005) also work on diacritization with the goal of improving ASR. They use a word-based language model (using both diacritized and undiacritized words in the context) but back off to a character-based model for unseen words. They consult BAMA to narrow possible diacritizations for unseen words, but BAMA does not provide much improvement used in this manner.

Zitouni et al. (2006) use a maximum entropy classifier to assign a set of diacritics to the letters of each word. They use the output of a tokenizer (segmenter) and a part-of-speech tagger (which presumably tags the output of the tokenizer). They then use segment n-grams, segment position of the character being diacritized, the POS of the current segment, along with lexical features, including letter and word n-grams. Thus, while many of the same elements are used in their and our work (word n-grams, features related to morphological analysis), the basic approach is quite different: while we have one procedure that chooses a correct analysis (including tokenization, morphological tag, and diacritization), they have a pipeline of processors. Furthermore, Zitouni et al. (2006) do not use a morphological lexicon. To our knowledge, their system is the best performing currently, and we have set up our experiments to allow us to compare our results directly to their results.

	All Diacritics		Ignore Last	
Model	**WER**	**DER**	**WER**	**DER**
Only-DLM-1	39.4	14.5	13.8	6.6
Tagger-DLM-1	15.9	5.3	6.2	2.5
Tagger-DLM-2	15.2	5.1	5.8	2.4
Tagger-DLM-3	15.1	5.0	5.7	2.4
Tagger-LLM-1	16.0	5.3	6.3	2.6
Tagger-LLM-2	15.0	4.9	5.6	2.2
Tagger-LLM-3	**14.9**	**4.8**	**5.5**	**2.2**
Only-LLM-3	35.5	10.8	8.8	3.6
Tagger-noLM	16.0	5.3	6.3	2.6
Zitouni	18.0	5.5	7.9	2.5

Figure 1: Diacritization Results (all followed by single-choice-diac model); our best results are shown in boldface; Only-DLM-1 is the baseline; "Zitouni" is (Zitouni et al., 2006)

5 Results

There are several ways of defining metrics for diacritization. In order to assure maximal comparability with the work of Zitouni et al. (2006), we adopt their metric.[5] We count all words, including numbers and punctuation. Each letter (or digit) in a word is a potential host for a set of diacritics; we count all diacritics on a single letter as a single binary choice. So, for example, if we correctly predict a shadda but get the vowel wrong, it counts as a wrong choice. We approximate non-variant diacritization by removing all diacritics from the final letter (**Ignore Last**), while counting that letter in the evaluation. We give diacritic error rate (**DER**) which tells us for how many letters we incorrectly restored all diacritics, and word error rate (**WER**), which tells us how many words had at least one DER.

The results are shown in Figure 1. Going top to bottom, we first see the baseline, **Only-DLM-1**, which is simply a diacritization unigram model with

[5] We thank Imed Zitouni (personal communication) for details on their evaluation.

random choice for unseen words. We then show the results using the morphological tagger along with a language model. We first show results for the diacritization model, with 1-, 2-, and 3-grams. As we can see, the bigram language model helps slightly. The next three lines are the three lexeme n-gram models. Here we see that the unigram model performs worse than the unigram diacritization model, while the bigram and trigram models perform better (the trigram lexeme model is our best result). We interpret this as meaning that the lexeme model is useful only when context is taken into account, because it is actually performing a rudimentary form of WSD. We tease apart the contribution of the tagger and of the language model with two further experiments, in the next two lines: using just the lexeme language model (trigrams), and running just the tagger, followed by random choice. We can see that the tagger alone does as well as the tagger with the unigram lexeme model, while the lexeme model on its own does much worse. However, as expected, the lexeme model on its own for the Ignore Last measure is much closer to the performance of the tagger on its own. We conclude from this that the quite simple lexeme model is in fact contributing to the correct choice of the lexemic diacritics. Finally, we give the results of Zitouni et al. (2006) on the last line, which we understand to be the best published results currently. We see that we improve on their results in all categories. We can see the effect of our different approaches to diacritization in the numbers: while for WER we reduce the Zitouni et al error by 17.2%, the DER error reduction is only 10.9%. This is because we are choosing among complete diacritization options for white space-tokenized words, while Zitouni et al. (2006) make choices for each diacritic. This means that when we make a mistake, it may well affect several diacritics at once, so that the diacritic errors are concentrated in fewer words. This effect is even stronger when we disregard the final letter (30.4% reduction in WER versus 12.0% reduction in DER), suggesting that singleton errors in words tend to be in the final position (case, mood), as it is often hard for the tagger to determine these features.

6 Conclusion

We have shown that a diacritizer that uses a lexical resource can outperform a highly optimized *ad-hoc* diacritization system that draws on a large number of features. We speculate that further work on WSD could further improve our results. We also note the issue of unknown words, which will affect our system much more than that of (Zitouni et al., 2006). It is possible to construct a combined system which uses a lexicon, but backs off to a Zitouni-style system for unknown words. However, a large portion of the unknown words are in fact foreign words and names, and it is not clear whether the models learned handle such words well.

References

S. Ananthakrishnan, S. Narayanan, and S. Bangalore. 2005. Automatic diacritization of arabic transcripts for asr. In *Proceedings of ICON-05*, Kanpur, India.

T. Buckwalter. 2004. Buckwalter Arabic morphological analyzer version 2.0.

J. Giménez and L. Màrquez. 2004. Svmtool: A general pos tagger generator based on support vector machines. In *Proceedings of LREC'04*.

N. Habash and O. Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of (ACL'05)*.

N. Habash and O. Rambow. 2007. Arabic Diacritization through Full Morphological Tagging: A Detailed Discussion. Techncial Report, Center for Computational Learning Systems, Columbia University.

Jan Hajič, Otakar Smrž, Tim Buckwalter, and Hubert Jin. 2005. Feature-based tagger of approximations of functional Arabic morphology. In *Proceedings of the Workshop on Treebanks and Linguistic Theories (TLT)*, Barcelona, Spain.

Jan Hajič. 2000. Morphological tagging: Data vs. dictionaries. In *Proceedings of (NAACL'00)*.

Taku Kudo and Yuji Matsumoto. 2003. Fast methods for kernel-based text analysis. In *Proceedings of (ACL'03)*.

Mohamed Maamouri, Ann Bies, and Tim Buckwalter. 2004. The Penn Arabic Treebank: Building a large-scale annotated arabic corpus. In *NEMLAR Conference on Arabic Language Resources and Tools*, Cairo, Egypt.

Mohamed Maamouri, Ann Bies, and Seth Kulick. 2006. Diacritization: A challenge to arabic treebank annotation and parsing. In *Proceedings of the Conference of the Machine Translation SIG of the British Computer Society*.

Noah A. Smith, David A. Smith, and Roy W. Tromble. 2005. Context-based morphological disambiguation with random fields. In *Proceedings of (EMNLP05)*.

Andreas Stolcke. 2002. SRILM - an Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*.

Dimitra Vergyri and Katrin Kirchhoff. 2004. Automatic diacritization of arabic for acoustic modeling in speech recognition. In *COLING 2004 Workshop on Computational Approaches to Arabic Script-based Languages*, Geneva, Switzerland.

Imed Zitouni, Jeffrey S. Sorensen, and Ruhi Sarikaya. 2006. Maximum entropy based restoration of arabic diacritics. In *Proceedings of ACL'06*.

Are Very Large N-best Lists Useful for SMT?

Saša Hasan, Richard Zens, Hermann Ney
Human Language Technology and Pattern Recognition
Lehrstuhl für Informatik 6 – Computer Science Department
RWTH Aachen University, D-52056 Aachen, Germany
{hasan,zens,ney}@cs.rwth-aachen.de

Abstract

This paper describes an efficient method to extract large n-best lists from a word graph produced by a statistical machine translation system. The extraction is based on the k shortest paths algorithm which is efficient even for very large k. We show that, although we can generate large amounts of distinct translation hypotheses, these numerous candidates are not able to significantly improve overall system performance. We conclude that large n-best lists would benefit from better discriminating models.

1 Introduction

This paper investigates the properties of large n-best lists in the context of statistical machine translation (SMT). We present a method that allows for fast extraction of very large n-best lists based on the k shortest paths algorithm by (Eppstein, 1998). We will argue that, despite being able to generate a much larger amount of hypotheses than previously reported in the literature, there is no significant gain of such a method in terms of translation quality.

In recent years, phrase-based approaches evolved as the dominating method for feasible machine translation systems. Many research groups use a decoder based on a log-linear approach incorporating phrases as main paradigm (Koehn et al., 2003). As a by-product of the decoding process, one can extract n-best translations from a word graph and use these fully generated hypotheses for additional reranking.

In the past, several groups report on using n-best lists with n ranging from 1 000 to 10 000. The advantage of n-best reranking is clear: we can apply complex reranking techniques, based e.g. on syntactic analyses of the candidates or using huge additional language models, since the whole sentence is already generated. During the generation process, these models would either need hard-to-implement algorithms or large memory requirements.

1.1 Related work

The idea of n-best list extraction from a word graph for SMT was presented in (Ueffing et al., 2002). In (Zens and Ney, 2005), an improved method is reported that overcomes some shortcomings, such as duplicate removal by determinization of the word graph (represented as a weighted finite state automaton) and efficient rest-cost estimation with linear time complexity.

There are several research groups that use a two-pass approach in their MT systems. First, they generate n-best translation hypotheses with the decoder. Second, they apply additional models to the output and rerank the candidates (see e.g. (Chen et al., 2006)).

Syntactic features were investigated in (Och et al., 2004) with moderate success. Although complex models, such as features based on shallow parsing or treebank-based syntactic analyses, were applied to the n-best candidates, the "simpler" ones were more promising (e.g. IBM model 1 on sentence-level).

In the following section 2, we describe our SMT system and explain how an improved n-best extraction method is capable of generating a very large number of distinct candidates from the word graph. In section 3, we show our experiments related to n-best list reranking with various sizes and the corresponding performance in terms of MT evaluation measures. Finally, we discuss the results in section 4 and give some conclusive remarks.

Proceedings of NAACL HLT 2007, Companion Volume, pages 57–60,
Rochester, NY, April 2007.

2 Generating N-best lists

We use a phrase-based SMT system (Mauser et al., 2006) and enhance the n-best list extraction with Eppstein's k shortest path algorithm which allows for generating a very large number of translation candidates in an efficient way.

2.1 Baseline SMT system

The baseline system uses phrases automatically extracted from a word-aligned corpus (trained with GIZA++) and generates the best translations using weighted log-linear model combination with several features, such as word lexicon, phrase translation and language models. This direct approach is currently used by most state-of-the-art decoders. The model scaling factors are trained discriminatively on some evaluation measure, e.g. BLEU or WER, using the simplex method.

2.2 N-best list extraction

We incorporated an efficient extraction of n best translations using the k shortest path algorithm (Eppstein, 1998) into a state-of-the-art SMT system. The implementation is partly based on code that is publicly available.[1]

Starting point for the extraction is a word graph, generated separately by the decoder for each sentence. Since these word graphs are directed and acyclic, it is possible to construct a shortest path tree spanning from the sentence begin node to the end node. The efficiency of finding the k shortest paths in this tree lies in the book-keeping of edges through a binary heap that allows for an implicit representation of paths. The overall performance of the algorithm is efficient even for large k. Thus, it is feasible to use in situations where we want to generate a large number of paths, i.e. translation hypotheses in this context.

There is another issue that has to be addressed. In phrase-based SMT, we have to deal with different phrase segmentations for each sentence. Due to the large number of phrases, it is possible that we have paths through the word graph representing the same sentence but internally having different phrase boundaries. In n-best list generation, we want to get rid of these duplicates. Due to the efficiency of the k shortest paths algorithm, we allow for generating a very large number of hypotheses (e.g. $100 \cdot n$) and then filter the output via a prefix tree (also called trie) until we get n distinct translations.

With this method, it is feasible to generate 100 000-best lists without much hassle. In general, the file input/output operations are more time-consuming than the actual n-best list extraction. The average generation time of n-best candidates for each of the sentences of the development list is approximately 30 seconds on a 2.2GHz Opteron machine, whereas 7.4 million hypotheses are computed per sentence on average. The overall extraction time including filtering and writing to hard-disk takes around 100 seconds per sentence. Note that this value could be optimized drastically if checking for how many duplicates are generated on average beforehand and adjusting the initial number of hypotheses before applying the filtering. We only use the $k = 100 \cdot n$ as a proof of concept.

2.3 Rescoring models

After having generated the 100 000-best lists, we have to apply additional rescoring models to all hypotheses. We select the models that have shown to improve overall translation performance as used for recent NIST MT evaluations. In addition to the main decoder score (which is already a combination of several models and constitutes a strong baseline), these include several large language models trained on up to 2.5 billion running words, a sentence-level IBM model 1 score, m-gram posterior probabilities and an additional sentence length model.

3 Experiments

The experiments in this section are carried out on n-best lists with n going up to 100 000. We will show that, although we are capable of generating this large amount of hypotheses, the overall performance does not seem to improve significantly beyond a certain threshold. Or to put it simple: although we generate lots of hypotheses, most of them are not very useful.

As experimental background, we choose the large data track of the Chinese-to-English NIST task, since the length of the sentences and the large vocabulary of the task allow for large n-best lists. For smaller tasks, e.g. the IWSLT campaign, the domain is rather limited such that it does not make sense to generate lists reaching beyond several thousand hypotheses. As development data, we use the 2002 eval set, whereas for test, the 2005 eval set is chosen. The corpus statistics are shown in Table 1.

[1] `http://www.ics.uci.edu/~eppstein/pubs/graehl.zip`

		Chinese	English
Train	Sentence Pairs	7M	
	Running Words	199M	213M
	Vocabulary Size	222K	351K
Dev	Sentence Pairs	878	3 512
	Running Words	25K	105K
Test	Sentence Pairs	1 082	4 328
	Running Words	33K	148K

Table 1: Corpus statistics for the Chinese-English NIST MT task.

	Oracle-WER [%]		Oracle-PER [%]	
N	BLEU	abs. imp.	BLEU	abs. imp.
1	36.1		36.1	
10	38.8	+2.7	38.0	+1.9
100	41.3	+2.5	39.8	+1.8
1000	43.3	+2.0	41.0	+1.2
10000	44.4	+1.1	42.0	+1.0
100000	45.3	+0.9	42.0	+0.0

Table 2: Dev BLEU scores of oracle-best hypotheses based on minimum WER/PER.

3.1 Oracle-best hypotheses

In the first experiment, we examined the oracle-best hypotheses in the n-best lists for several list sizes. For an efficient calculation of the true BLEU oracle (the hypothesis which has a maximum BLEU score when compared to the reference translations), we use approximations based on WER/PER-oracles, i.e. we extract the hypotheses that have the lowest edit distance (WER, word error rate) to the references. The same is applied by disregarding the word order (leading to PER, position-independent word error rate).

As can be seen in Table 2, the improvements are steadily decreasing, i.e. with increasing number of generated hypotheses, there are less and less useful candidates among them. For the first 10 000 candidates, we therefore have the possibility to find hypotheses that could increase the BLEU score by at least 8.3% absolute *if* our models discriminated them properly. For the next 90 000 hypotheses, there is only a small potential to improve the whole system by around 1%. This means that most of the generated hypotheses are not very useful in terms of oracle-WER and likely distracting the "search" for the needle(s) in the haystack. It has been shown in (Och et al., 2004) that true BLEU oracle scores on lists with much smaller $n \leq 4096$ are more or less linear in $\log(n)$. Our results support this claim since the oracle-WER/PER is a lower bound of the real BLEU oracle. For the PER criterion, the behavior of the oracle-best hypotheses is similar. Here we can notice that after 10,000 hypotheses, the BLEU score of the oracle-PER hypotheses stays the same.

These observations already impair the alleged usefulness of a large amount of translation hypotheses by showing that the overall possible gain with increasing n gets disproportionately small if one puts it in relation to the exponential growth of the n.

3.2 Rescoring performance

As a next step, we show the performance of tuning the model scaling factors towards best translation performance. In our experiments, we use the BLEU score as objective function of the simplex method.

Figure 1 shows the graphs for the development (on the left) and test set (on the right). The upper graphs depict the oracle-WER BLEU scores (cf. also Table 2) for comparison. As was already stated, these are a lower bound since the real oracle-BLEU hypotheses might have even higher scores. Still, it is an indicator of what could be achieved if the models discriminated good from bad hypotheses properly.

The lower two graphs show the behavior when (a) optimizing and extracting hypotheses on a subset (the first n) of the 100k-best hypotheses and (b) optimizing on a subset but extracting from the full 100k set. As can be seen, extracting from the full set does not even help for the development data on which the scaling factors were tuned. Experiments on the test list show similar results. We can also observe that the improvement declines rapidly with higher n. Note that an optimization on the full 100k list was not possible due to huge memory requirements. The highest n that fit into the 16GB machine was 60 000. Thus, this setting was used for extraction on the full 100k set.

The results so far indicate that it is not very useful to go beyond $n = 10\,000$. For the development set, the baseline of 36.1% BLEU can be improved by 1.6% absolute to 37.7% for the first 10k entries, whereas for the 60k setting, the absolute improvement is only increased by a marginal 0.1%. For the chosen setting, whose focus was on various list sizes for optimization and extraction, the improvements on the development lists do not carry over to the test list. From the baseline of 31.5%, we only get a moderate improvement of approximately 0.5% BLEU.

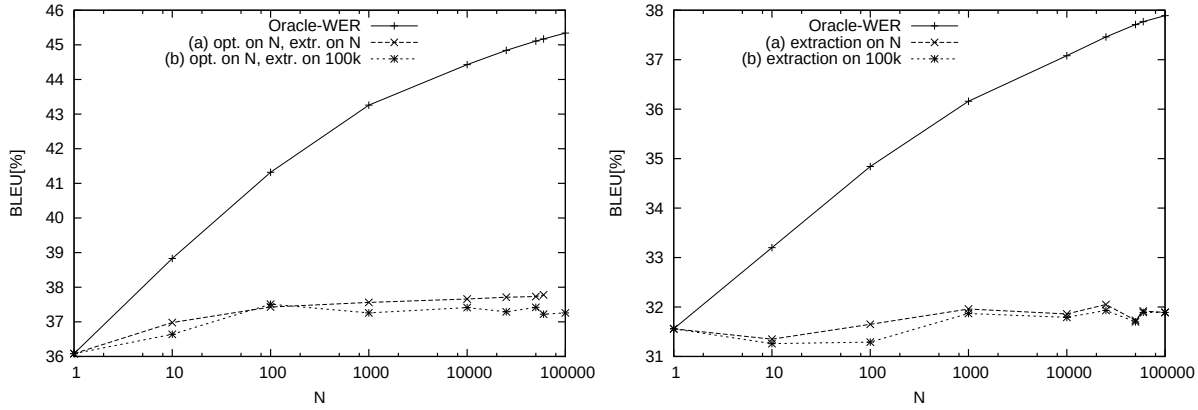


Figure 1: BLEU scores of the reranked system. Development set (left) vs. Test set (right).

One possible explanation for this lies in the poor performance of the rescoring models. A short test was carried out in which we added the reference translations to the n-best list and determined the corresponding scores of the additional models, such as the large LM and the IBM model 1. Interestingly, only less than 1/4 of the references was ranked as the best hypothesis. Thus, most reference translations would never have been selected as final candidates. This strongly indicates that we have to come up with better models in order to make significant improvements from large n-best lists. Furthermore, it seems that the exponential growth of n-best hypotheses for maintaining a quasilinear improvement in oracle BLEU score has a strong impact on the overall system performance. This is in contrast to a word graph, where a linear increment of its density yields disproportionately high improvements in oracle BLEU for lower densities (Zens and Ney, 2005).

4 Conclusion

We described an efficient n-best list extraction method that is based on the k shortest paths algorithm. Experiments with large 100 000-best lists indicate that the models do not have the discriminating power to separate the good from the bad candidates. The oracle-best BLEU scores stay linear in $\log(n)$, whereas the reranked system performance seems to saturate at around 10k best translations given the actual models. Using more hypotheses currently does not help to significantly improve translation quality.

Given the current results, one should balance the advantages of n-best lists, e.g. easily testing complex rescoring models, and word graphs, e.g. representation of a much larger hypotheses space. However, as long as the models are not able to correctly fire on good candidates, both approaches will stay beneath their capabilities.

Acknowledgments

This material is partly based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023, and was partly funded by the Deutsche Forschungsgemeinschaft (DFG) under the project "Statistische Textübersetzung" (NE 572/5-3).

References

B. Chen, R. Cattoni, N. Bertoldi, M. Cettolo, and M. Federico. 2006. The ITC-irst SMT system for IWSLT 2006. In *Proc. of the International Workshop on Spoken Language Translation*, pages 53–58, Kyoto, Japan, November.

D. Eppstein. 1998. Finding the k shortest paths. *SIAM J. Computing*, 28(2):652–673.

P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proc. of the Human Language Technology Conf. (HLT-NAACL)*, pages 127–133, Edmonton, Canada, May/June.

A. Mauser, R. Zens, E. Matusov, S. Hasan, and H. Ney. 2006. The RWTH statistical machine translation system for the IWSLT 2006 evaluation. In *Proc. of the International Workshop on Spoken Language Translation*, pages 103–110, Kyoto, Japan, November.

F. J. Och, D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin, and D. Radev. 2004. A smorgasbord of features for statistical machine translation. In *Proc. 2004 Meeting of the North American chapter of the Association for Computational Linguistics (HLT-NAACL)*, pages 161–168, Boston, MA, May.

N. Ueffing, F. J. Och, and H. Ney. 2002. Generation of word graphs in statistical machine translation. In *Proc. of the Conf. on Empirical Methods for Natural Language Processing (EMNLP)*, pages 156–163, Philadelphia, PA, July.

R. Zens and H. Ney. 2005. Word graphs for statistical machine translation. In *43rd Annual Meeting of the Assoc. for Computational Linguistics: Proc. Workshop on Building and Using Parallel Texts*, pages 191–198, Ann Arbor, MI, June.

Relationship between Non-Projective Edges, their Level Types, and Well-Nestedness

Jiří Havelka
Institute of Formal and Applied Linguistics
Charles University in Prague, Czech Republic
havelka@ufal.mff.cuni.cz

Abstract

Dependency analysis of natural language gives rise to non-projective structures. The constraint of well-nestedness on dependency trees has been recently shown to give a good fit with empirical linguistic data. We present a reformulation of this constraint using properties of non-projective edges and show its formal relationship to level types of non-projective edges; we also derive a simple $O(n^2)$ algorithm for checking well-nestedness.

1 Introduction

Dependency analysis of natural language has been gaining an ever increasing interest thanks to its applicability in many tasks of NLP—a recent example is the dependency parsing work of McDonald et al. (2005), which introduces an approach based on the search for maximum spanning trees, capable of handling non-projective structures naturally.

In this context, the issue of delimiting dependency structures permissible in natural language grows in importance (see e.g. Nivre (2006)). We would like to point out that since neither syntactic structures in dependency treebanks, nor structures arising in dependency parsing need a priori fall into any formal subclass of dependency trees, we need means of describing any non-projective structure.[1]

Kuhlmann and Nivre (2006) compare several constraints on dependency structures and among the considered ones find well-nestedness to be in good accord with empirical data. However, they do not include level types of non-projective edges introduced by Havelka (2005), which present another means of characterizing any non-projective structure and have interesting formal properties. We link properties of non-projective edges and their level types to the constraint of well-nestedness and show that they provide a more fine-grained means capable of capturing it.

The paper is organized as follows: Sect. 2 contains formal preliminaries; Sect. 3 and 4 review definitions and show the necessary properties of the constraint of well-nestedness and level types of non-projective edges; Sect. 5 presents the main results concerning the relationship between non-projective edges (and their level types) and well-nestedness.

2 Formal preliminaries

To make the paper as self-contained as possible, we provide a concise reference with definitions and simple properties used in subsequent sections.

Definition 1 A *dependency tree* is a triple $(V, \rightarrow, \preceq)$, where V is a finite set of nodes, $\rightarrow$ a *dependency* relation on V, and $\preceq$ a total order on V.

Relation $\rightarrow$ models linguistic dependency, and so represents a directed, rooted tree on V. There are many ways of characterizing rooted trees, we give here a characterization via the properties of $\rightarrow$: there is a *root* $r \in V$ such that $r \rightarrow^* v$ for all $v \in V$ and there

[1] The importance of such means is evident from the asymptotically negligible proportion of projective trees to all dependency trees. (Unrestricted dep. trees (i.e. labelled rooted trees), well-nested dep. trees, and projective dep. trees are counted by sequences A000169, A113882, and A006013 (offset 1), resp., in the On-Line Encyclopedia of Sequences (Sloane, 2007).)

Proceedings of NAACL HLT 2007, Companion Volume, pages 61–64,
Rochester, NY, April 2007.

is a unique *edge* $p \to v$ for all $v \in V$, $v \neq r$. Relation $\to^*$ is the reflexive transitive closure of $\to$ and is usually called *subordination*.

The following definitions allow us to formulate our results succinctly. For each node i we define its *level* as the length of the path $r \to^* i$; we denote it level_i. The symmetrization $\leftrightarrow = \to \cup \to^{-1}$ makes it possible to talk about edges (pairs of nodes i, j such that $i \to j$) without explicitly specifying the *parent* (*head* etc.; i here) and the *child* (*dependent* etc.; j here); so $\to$ represents directed edges and $\leftrightarrow$ undirected edges. To retain the ability to talk about the direction of edges, we define $\mathsf{Parent}_{i\leftrightarrow j} = \begin{cases} i & \text{if } i \to j \\ j & \text{if } j \to i \end{cases}$ and $\mathsf{Child}_{i\leftrightarrow j} = \begin{cases} j & \text{if } i \to j \\ i & \text{if } j \to i \end{cases}$. Our notation for rooted *subtrees* is $\mathsf{Subtree}_i = \{v \in V \mid i \to^* v\}$, $\mathsf{Subtree}_{i\leftrightarrow j} = \{v \in V \mid \mathsf{Parent}_{i\leftrightarrow j} \to^* v\}$, and for *ancestors* $\mathsf{Anc}_i = \{v \in V \mid v \to^* i\}$, $\mathsf{Anc}_{i\leftrightarrow j} = \{v \in V \mid v \to^* \mathsf{Parent}_{i\leftrightarrow j}\}$. To be able to talk concisely about the total order on nodes $\preceq$, we define *open* and *closed intervals* whose endpoints need not be in a prescribed order: $(i, j) = \{v \in V \mid \min_{\preceq}\{i, j\} \prec v \prec \max_{\preceq}\{i, j\}\}$ and $[i, j] = \{v \in V \mid \min_{\preceq}\{i, j\} \preceq v \preceq \max_{\preceq}\{i, j\}\}$, resp. For any edge $i \leftrightarrow j$ we define its *gap* as follows $\mathsf{Gap}_{i\leftrightarrow j} = \{v \in V \mid v \in (i, j) \ \&\ v \notin \mathsf{Subtree}_{i\leftrightarrow j}\}$. An edge with an empty gap is *projective*, an edge whose gap is non-empty is *non-projective*. (See e.g. (Havelka, 2005) for the characterization of projectivity via properties of edges and further references.)

Property 2 *Let a be a node and $i \leftrightarrow j$ any edge disjoint from a. Then $i \in \mathsf{Subtree}_a \Leftrightarrow j \in \mathsf{Subtree}_a$.*
PROOF. From the assumption $i \neq a \neq j$ it follows that $i, j \in \mathsf{Subtree}_a \Leftrightarrow \mathsf{Parent}_{i\leftrightarrow j} \in \mathsf{Subtree}_a$. ∎

Proposition 3 *Let $i \leftrightarrow j$, $u \leftrightarrow v$ be disjoint edges.*
(i) *If $u, v \in (i, j)$, then $u \in \mathsf{Gap}_{i\leftrightarrow j} \Leftrightarrow v \in \mathsf{Gap}_{i\leftrightarrow j}$.*
(ii) *If $u \in \mathsf{Gap}_{i\leftrightarrow j}$ and $v \notin \mathsf{Gap}_{i\leftrightarrow j}$, then $v \notin [i, j]$.*
PROOF. (i) follows immediately from the definition of $\mathsf{Gap}_{i\leftrightarrow j}$ and Property 2. To prove (ii), assume $v \in (i, j)$ and using (i) arrive at a contradiction. ∎

3 Well-nestedness

Kuhlmann and Nivre (2006) claim that the constraint of well-nestedness seems to approximate well dependency structures occurring in natural language.

Definition 4 A dependency tree T is *ill-nested* if there are disjoint subtrees T_1, T_2 of T and nodes $x_1, y_1 \in T_1$ and $x_2, y_2 \in T_2$ such that $x_1 \in (x_2, y_2)$ and $x_2 \in (x_1, y_1)$. A dependency tree T that is not ill-nested is *well-nested*.[2]

It is easy to express the constraint in terms of edges—it will prove crucial in Sect. 5.

Theorem 5 *A dependency tree T is ill-nested iff there are edges $i_1 \leftrightarrow j_1$, $i_2 \leftrightarrow j_2$ in disjoint subtrees T_1, T_2 of T, resp., such that $i_1 \in (i_2, j_2)$, $i_2 \in (i_1, j_1)$.*
PROOF. Direction $\Leftarrow$ is obvious.

Direction $\Rightarrow$: Let r_i be the root of T_i. To find $i_1 \leftrightarrow j_1$, first suppose that $r_1 \in (x_2, y_2)$. Consider the first edge $v_k \to v_{k+1}$ on the downward path $v_0 = r_1$, $v_1, \ldots, v_m = y_1$, $m > 0$, such that $v_k \in (x_2, y_2)$ and $v_{k+1} \notin [x_2, y_2]$. If $r_1 \notin [x_2, y_2]$, consider the first edge $v_{k+1} \to v_k$ on the upward path $v_0 = x_1, v_1, \ldots, v_n = r_1$, $n > 0$, such that $v_k \in (x_2, y_2)$ and $v_{k+1} \notin [x_2, y_2]$. Let us denote $i_1 = v_k$ and $j_1 = v_{k+1}$, and possibly rename x_2, y_2 so that $i_1 \in (x_2, y_2)$ and $x_2 \in (i_1, j_1)$. To find $i_2 \leftrightarrow j_2$ such that $i_1 \in (i_2, j_2)$, $i_2 \in (i_1, j_1)$, proceed similarly as above. Obviously, edges $i_1 \leftrightarrow j_1$, $i_2 \leftrightarrow j_2$ are in disjoint subtrees. ∎

4 Level types of non-projective edges

Level types of non-projective edges allow their structural classification with interesting formal properties. They were introduced by Havelka (2005), who presents them in more detail.

Definition 6 The *level type* (or just *type*) of a non-projective edge $i \leftrightarrow j$ is defined as follows

$$\mathsf{Type}_{i\leftrightarrow j} = \mathsf{level}_{\mathsf{Child}_{i\leftrightarrow j}} - \min_{n \in \mathsf{Gap}_{i\leftrightarrow j}} \mathsf{level}_n \ .$$

The type of an edge is the distance of its child node and a node in its gap closest to the root (distance here means difference in levels)—for sample configurations see Figure 1[3]. Note that there may be more than one node witnessing an edge's type. The type of an edge is not bounded—it can take any integer value (depending on the height of a tree).

Our definition of level type of non-projective edges extends naturally the original definition im-

[2] The original definition requires $x_1 \prec x_2 \prec y_1 \prec y_2$, however our equivalent formulation leads to shorter theorems and proofs.

[3] We adopt the following convention: nodes are drawn top-down according to their increasing level, with nodes on the same level on the same horizontal line; nodes are drawn from left to right according to the total order on nodes; edges are drawn as solid lines, paths as dotted curves. We assume that no node on a path lies in the span of an edge the path crosses.

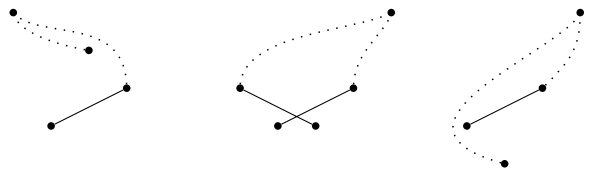


Figure 1: Sample non-projective edges

plicit in (Havelka, 2005), where classes corresponding to positive, zero and negative types are defined.

We now show a relationship between edges of certain types that will allow us to derive a relationship between non-projective edges and well-nestedness.

Theorem 7 *For any non-projective edge $i \leftrightarrow j$ in a dependency tree T with $\mathsf{Type}_{i\leftrightarrow j} \leq 0$ (< 0) there is a non-projective edge $v \rightarrow u$ in T with $\mathsf{Type}_{u\leftrightarrow v} \geq 0$ (> 0) such that $u \in \arg\min_{n\in\mathsf{Gap}_{i\leftrightarrow j}} \mathsf{level}_n$ and either $i \in \mathsf{Gap}_{u\leftrightarrow v}$, or $j \in \mathsf{Gap}_{u\leftrightarrow v}$.*

PROOF. Let u be any node in $\arg\min_{n\in\mathsf{Gap}_{i\leftrightarrow j}} \mathsf{level}_n$. From the assumption $\mathsf{Type}_{i\leftrightarrow j} \leq 0$ node u has a parent $v \notin \mathsf{Gap}_{i\leftrightarrow j}$. Obviously $i \leftrightarrow j$, $v \rightarrow u$ are disjoint, thus from Proposition 3 we have $v \notin [i, j]$, and so either $i \in (u, v)$, or $j \in (u, v)$. Since $\mathsf{level}_v \geq \mathsf{level}_{\mathsf{Parent}_{i\leftrightarrow j}}$, we have that $\mathsf{Parent}_{i\leftrightarrow j} \notin \mathsf{Subtree}_v$, and so either $i \in \mathsf{Gap}_{u\leftrightarrow v}$, or $j \in \mathsf{Gap}_{u\leftrightarrow v}$. Finally from $\mathsf{Type}_{i\leftrightarrow j} \leq 0$ (< 0) we get $\mathsf{level}_u - \mathsf{level}_{\mathsf{Child}_{i\leftrightarrow j}} \geq 0$ (> 0), hence $\mathsf{Type}_{u\leftrightarrow v} \geq 0$ (> 0). ∎

5 Well-nestedness & non-projective edges

We give characterizations of well-nestedness solely in terms of properties of non-projective edges and show some applications.

5.1 Characterization using pairs of edges

First we give a characterization of pairs of edges in Theorem 5 in terms of their gaps.

Theorem 8 *Let $i_1 \leftrightarrow j_1$, $i_2 \leftrightarrow j_2$ be two edges in a dependency tree T. They are in disjoint subtrees T_1, T_2, resp., and satisfy $i_1 \in (i_2, j_2)$, $i_2 \in (i_1, j_1)$ iff the following condition holds*

$$\text{(inp)} \qquad i_1 \in \mathsf{Gap}_{i_2\leftrightarrow j_2} \;\;\&\;\; i_2 \in \mathsf{Gap}_{i_1\leftrightarrow j_1} \;.$$

PROOF. Direction $\Leftarrow$: Root T_k in $\mathsf{Parent}_{i_k\leftrightarrow j_k}$, $k = 1, 2$. Condition (inp) obviously implies $i_1 \in (i_2, j_2)$, $i_2 \in (i_1, j_1)$, which in turn implies that edges $i_1 \leftrightarrow j_1$, $i_2 \leftrightarrow j_2$ are disjoint. From Property 2 we get that both $\mathsf{Parent}_{i_2\leftrightarrow j_2} \notin \mathsf{Subtree}_{i_1\leftrightarrow j_1}$ and $\mathsf{Parent}_{i_1\leftrightarrow j_1} \notin \mathsf{Subtree}_{i_2\leftrightarrow j_2}$, hence subtrees T_1, T_2 are disjoint.

Direction $\Rightarrow$: Let us consider the edge $i_2 \leftrightarrow j_2$ and node i_1. Since T_1 is disjoint from T_2, we have that $i_1 \notin \mathsf{Subtree}_{i_2\leftrightarrow j_2}$, and therefore $i_1 \in \mathsf{Gap}_{i_2\leftrightarrow j_2}$. The proof that $i_2 \in \mathsf{Gap}_{i_1\leftrightarrow j_1}$ is analogous. ∎

Condition (inp) allows us to talk about pairs of edges causing ill-nestedness and so characterize well-nestedness using properties of pairs of edges.

Definition 9 We say that any two non-projective edges $i_1 \leftrightarrow j_1$, $i_2 \leftrightarrow j_2$ in a dependency tree T satisfying condition (inp) form an *ill-nested pair of edges*.

Corollary 10 *A dependency tree T is ill-nested iff it contains an ill-nested pair of edges.*

PROOF. Follows from Theorems 5 and 8. ∎

5.2 Sufficient condition for ill-nestedness

The results of Section 4 and previous subsection give the following relationship between types of non-projective edges and well-nestedness.

Theorem 11 *If a dependency tree contains a non-proj. edge of nonpositive type, then it is ill-nested.*

PROOF. Follows from Theorems 7 and 10. ∎

We see that types of non-projective edges and well-nestedness share a common ground; however, the statement of Theorem 11 cannot be strengthened to equivalence (it is easy to see that also two edges of positive type can satisfy (inp)).

5.3 Characterization using single edges

Now we show that well-nestedness can be characterized in terms of properties of single non-projective edges only. We define the ill-nested set of an edge and show that it gives the desired characterization.

Definition 12 The *ill-nested set* of any edge $i \leftrightarrow j$ is defined as follows

$$\mathsf{In}_{i\leftrightarrow j} = \{u \leftrightarrow v \mid u \in \mathsf{Gap}_{i\leftrightarrow j} \;\&\; v \notin [i, j] \;\&\; u, v \notin \mathsf{Anc}_{i\leftrightarrow j}\} \;.$$

The next proposition exposes the relationship of edges in $\mathsf{In}_{i\leftrightarrow j}$ to the gap of $i \leftrightarrow j$.

Proposition 13 *For any edge $i \leftrightarrow j$ $\mathsf{In}_{i\leftrightarrow j} = \{u \leftrightarrow v \mid u \in \mathsf{Gap}_{i\leftrightarrow j} \;\&\; v \notin \mathsf{Gap}_{i\leftrightarrow j} \;\&\; u, v \notin \mathsf{Anc}_{i\leftrightarrow j}\}$.*

PROOF. The inclusion $\subseteq$ is obvious. The inclusion $\supseteq$ follows from Proposition 3 ($u \in \mathsf{Gap}_{i\leftrightarrow j}$ and $v \notin \mathsf{Anc}_{i\leftrightarrow j}$ imply that edges $i \leftrightarrow j$, $u \leftrightarrow v$ are disjoint). ∎

We are ready to formulate the main result of this section, which gives as corollary a characterization of well-nestedness using properties of single edges.

Theorem 14 *Let $i \leftrightarrow j$ be an edge in a dependency tree T. The edges that form an ill-nested pair with the edge $i \leftrightarrow j$ are exactly the edges in $\mathsf{In}_{i\leftrightarrow j}$.*

PROOF. Direction $\Rightarrow$: Let $u \leftrightarrow v$ be an edge forming an ill-nested pair with the edge $i \leftrightarrow j$, i.e. $i \in \mathsf{Gap}_{u\leftrightarrow v}$ and $u \in \mathsf{Gap}_{i\leftrightarrow j}$. This implies $i \in (u,v)$ and $u \in (i,j)$, which immediately gives $v \notin [i,j]$. Supposing $u \in \mathsf{Anc}_{i\leftrightarrow j}$ or $v \in \mathsf{Anc}_{i\leftrightarrow j}$ we get $i \in \mathsf{Subtree}_{u\leftrightarrow v}$, which is in contradiction with $i \in \mathsf{Gap}_{u\leftrightarrow v}$, and therefore $u,v \notin \mathsf{Anc}_{i\leftrightarrow j}$. Hence $u \leftrightarrow v \in \mathsf{In}_{i\leftrightarrow j}$.

Direction $\Leftarrow$: Let $u \leftrightarrow v \in \mathsf{In}_{i\leftrightarrow j}$ (i.e. $u \in \mathsf{Gap}_{i\leftrightarrow j}$, $v \notin [i,j]$, and $u,v \notin \mathsf{Anc}_{i\leftrightarrow j}$; without loss of generality assume $i \in (u,v)$). From the assumptions $u \in \mathsf{Gap}_{i\leftrightarrow j}$ and $v \notin [i,j]$ we get that edges $i \leftrightarrow j$, $u \leftrightarrow v$ are disjoint. Using Property 2, from the assumption $u,v \notin \mathsf{Anc}_{i\leftrightarrow j}$ we get $i \notin \mathsf{Subtree}_{u\leftrightarrow v}$, thus $i \in \mathsf{Gap}_{u\leftrightarrow v}$. Hence $i \leftrightarrow j$, $u \leftrightarrow v$ satisfy (inp). ∎

Corollary 15 *A dependency tree T is ill-nested iff $\mathsf{In}_{i\leftrightarrow j} \neq \emptyset$ for some non-projective edge $i \leftrightarrow j$ in T.*

PROOF. Follows from Theorems 8 and 14. ∎

5.4 Checking well-nestedness

Our characterization of well-nestedness gives also a novel way of checking it. Here is a pseudocode of an algorithm for fully determining all ill-nested sets:

```
1: for all edges i ↔ j do
2:     for all edges u ↔ v s.t. u ∈ (i, j) do
3:         check u ↔ v ∈ In_{i↔j}
```

Its time complexity is obviously $O(n^2)$, since the check on line 3 can be implemented so as to take constant time (by precompuing $\rightarrow^*$, which can be done in $O(n^2)$ time). The bound is the same as for the reported algorithms for checking well-nestedness (Möhl, 2006).

However, the following theorem allows well-nestedness checking to be linear for projective trees, to be faster for random input, and to remain $O(n^2)$.

Theorem 16 *In any ill-nested pair of edges, at least one of the edges is of nonnegative type (witnessed by an end-point of the other edge).*

PROOF. Let $i_1 \leftrightarrow j_1$, $i_2 \leftrightarrow j_2$ satisfy (inp). Let us suppose that $\mathsf{level}_{\mathsf{Child}_{i_1\leftrightarrow j_1}} \geq \mathsf{level}_{\mathsf{Child}_{i_2\leftrightarrow j_2}}$. Since $\mathsf{level}_{\mathsf{Child}_{u\leftrightarrow v}} \geq \mathsf{level}_u$ for any edge $u \leftrightarrow v$, we have that $\mathsf{level}_{\mathsf{Child}_{i_1\leftrightarrow j_1}} \geq \mathsf{level}_{i_2}$, and hence $\mathsf{Type}_{i_1\leftrightarrow j_1} \geq 0$. If $\mathsf{level}_{\mathsf{Child}_{i_1\leftrightarrow j_1}} \leq \mathsf{level}_{\mathsf{Child}_{i_2\leftrightarrow j_2}}$, it is analogously proved that $i_2 \leftrightarrow j_2$ is of nonnegative type. ∎

Havelka (2005) presents a linear algorithm for finding all non-projective edges of nonnegative type. Thus well-nestedness can be checked as follows: first find all edges of nonnegative type, and then check their ill-nested sets for non-emptiness. Computing $\rightarrow^*$ on demand for subtrees of the processed edges, we preserve worst-case quadratic complexity.

6 Conclusion

We have presented new formal results linking properties of non-projective edges and their level types to well-nestedness. This work extends the current body of research on non-projective dependency structures in natural language. In particular, we offer new insights into formal properties of non-projective edges that, if possible, both provide adequate means for linguistic description and at the same time are useful as features in machine-learning approaches.

Acknowledgement This work was supported by projects 1ET201120505 of the Ministry of Education of the Czech Republic and 374/2005/A-INF/MFF of Grant Agency of Charles University.

References

Jiří Havelka. 2005. Projectivity in Totally Ordered Rooted Trees: An Alternative Definition of Projectivity and Optimal Algorithms for Detecting Non-Projective Edges and Projectivizing Totally Ordered Rooted Trees. *Prague Bulletin of Mathematical Linguistics*, 84:13–30.

Marco Kuhlmann and Joakim Nivre. 2006. Mildly Non-Projective Dependency Structures. In *Proceedings of COLING/ACL*, pages 507–514.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-Projective Dependency Parsing using Spanning Tree Algorithms. In *Proceedings of HLT/EMNLP*, pages 523–530.

Mathias Möhl. 2006. Drawings as models of syntactic structure: Theory and algorithms. Diploma thesis, Programming Systems Lab, Universität des Saarlandes, Saarbrücken.

Joakim Nivre. 2006. Constraints on Non-Projective Dependency Parsing. In *Proc. of EACL*, pages 73–80.

Neil J. A. Sloane. 2007. On-Line Encyclopedia of Integer Sequences. Published electronically at `www.research.att.com/~njas/sequences/`.

iROVER: Improving System Combination with Classification

D. Hillard[†], B. Hoffmeister[‡], M. Ostendorf[†], R. Schlüter[‡], H. Ney[‡]
†SSLI, Electrical Engineering Dept., University of Washington, Seattle, WA
{hillard,mo}@ee.washington.edu
‡Informatik 6, Computer Science Dept., RWTH Aachen University, Aachen, Germany
{hoffmeister,schlueter,ney}@cs.rwth-aachen.de

Abstract

We present an improved system combination technique, iROVER. Our approach obtains significant improvements over ROVER, and is consistently better across varying numbers of component systems. A classifier is trained on features from the system lattices, and selects the final word hypothesis by learning cues to choose the system that is most likely to be correct at each word location. This approach achieves the best result published to date on the TC-STAR 2006 English speech recognition evaluation set.

1 Introduction

State-of-the-art automatic speech recognition (ASR) systems today usually include multiple contrasting systems, which are ultimately combined to produce the final hypothesis. There is consensus that improvements from combination are usually best when systems are sufficiently different, but there is uncertainty about which system combination method performs the best. In addition, the success of commonly used combination techniques varies depending on the number of systems that are combined (Hoffmeister et al., 2007). In this work, we develop a system combination method that outperforms all previously known techniques and is also robust to the number of component systems. The relative improvements over ROVER are particularly large for combination when only using two systems.

The aim of system combination for ASR is to minimize the expected word error rate (WER) given multiple system outputs, which are ideally annotated with word confidence information. The most widely used system combination approach to date is ROVER (Fiscus, 1997). It is a simple voting mechanism over just the top hypothesis from each component system. Two alternatives that incorporate information about multiple hypotheses and leverage word posterior probabilities are confusion network (CN) combination (Mangu et al., 2000; Evermann and Woodland, 2000) and minimum Time Frame Word Error (min-fWER) decoding (Hoffmeister et al., 2006), discussed further in the next section. Previous work found that among ROVER, CN combination, and min-fWER combination, no one method was consistently superior across varying numbers and types of systems (Hoffmeister et al., 2007).

The main contribution of this work is to develop an approach that always outperforms other possible system combination methods. We train a classifier to learn which system should be selected for each output word, using features that describe the characteristics of the component systems. ROVER alignments on the 1-best hypotheses are used for decoding, but many of the features are derived from the system lattices. The classifier learns a selection strategy (i.e. a decision function) from a development set and then is able to make better selections on the evaluation data then the current 1-best or lattice-based system combination approaches.

Next, Section 2 describes previous work in system combination techniques. Section 3 describes our approach, and Section 4 provides experiments and results. Finally, we summarize the approach and findings in Section 5.

2 Previous Work

Previous work in speech recognition system combination has produced significant improvements over the results possible with just a single system. The most popular, and often best performing method is ROVER (Fiscus, 1997), which selects the word that the most systems agree on at a particular location (majority voting). An extended version of ROVER also weights system votes by the word confidence produced by the system (confidence voting).

Further improvements have been achieved by including multiple system alternatives, with methods such as Confusion Network Combination (CNC) (Evermann and Woodland, 2000), or N-Best ROVER (Stolcke et al., 2000), which is a special case of CNC. Alternatively, the combination can be performed at the frame level (min-fWER) (Hoffmeister et al., 2006). Recent work found that the best system combination method depended on the number of systems being combined (Hoffmeister et al., 2007). When only two systems are available, approaches considering multiple alternatives per system were bet-

Proceedings of NAACL HLT 2007, Companion Volume, pages 65–68,
Rochester, NY, April 2007.

ter, but as the number of systems increased the standard ROVER with confidence scores was more robust and sometimes even better than CNC or min-fWER combination.

Another approach (Zhang and Rudnicky, 2006) used two stages of neural networks to select a system at each word, with features that capture word frequency, posteriors at the frame, word, and utterance level, LM back-off mode, and system accuracy. They obtained consistent but small improvements over ROVER: between 0.7 and 1.7% relative gains for systems with about 30% WER.

3 Approach

We develop a system that uses the ROVER alignment but learns to consistently make better decisions than those of standard ROVER. We call the new system *i*ROVER, where the *i* stands for improved results, and/or intelligent decisions. The following sections discuss the components of our approach. First, we emulate the approach of ROVER in our lattice preprocessing and system alignment. We then introduce new methods to extract hypothesis features and train a classifier that selects the best system at each slot in the alignment.

3.1 Lattice Preparation

Our experiments use lattice sets from four different sites. Naturally, these lattice sets differ in their vocabulary, segmentation, and density. A compatible vocabulary is essential for good combination performance. The main problems are related to contractions, e.g. "you've" and "you have", and the alternatives in writing foreign names, e.g. "Schröder" and "Schroder". In ASR this problem is well-known and is addressed in scoring by using mappings that allow alternative forms of the same word.

Such a mapping is provided within the TC-STAR Evaluation Campaign and we used it to normalize the lattices. In case of multiple alternative forms we used only the most frequent one. Allowing multiple parallel alternatives would have distorted the posterior probabilities derived from the lattice. Furthermore, we allowed only one-to-one or one-to-many mappings. In the latter case we distributed the time of the lattice arc according to the character lengths of the target words.

In order to create comparable posterior probabilities over the lattice sets we pruned them to equal average density. The least dense lattice set defined the target density: around 25 for the development and around 30 for the evaluation set.

Finally, we unified the segmentation by concatenating the lattices recording-wise. The concatenation was complicated by segmentations with overlapping regions, but our final concatenated lattices scored equally to the original lattice sets. The unified segmentation is needed for lattice-based system combination methods like frame-based combination.

3.2 System Alignments

In this work we decided to use the ROVER alignment as the basis for our system combination approach. At first glance the search space used by ROVER is very limited because only the first-best hypothesis from each component system is used. But the oracle error rate is often very low, normally less than half of the best system's error rate.

The ROVER alignment can be interpreted as a confusion network with an equal number of arcs in each slot. The number of arcs per slot equals the number of component systems and thus makes the training and application of a classifier straightforward.

For the production of the alignments we use a standard, dynamic programming-based matching algorithm that minimizes the global cost between two hypothesis. The local cost function is based on the time overlap of two words and is identical to the one used by the ROVER tool. We also did experiments with alternative local cost functions based on word equalities, but could not outperform the simple, time overlap-based distance function.

3.3 Hypothesis Features

We generate a cohort of features for each slot in the alignment, which is then used as input to train the classifier. The features incorporate knowledge about the scores from the original systems, as well as comparisons among each of the systems. The following paragraphs enumerate the six classes of feature types used in our experiments (with their names rendered in italics).

The primary, and most important feature class covers the *basic* set of features which indicate string matches among the top hypotheses from each system. In addition, we include the systems' frame-based word confidence. These features are all the information available to the standard ROVER with confidences voting.

An additional class of features provides extended *confidence* information about each system's hypothesis. This feature class includes the confusion network (CN) word confidence, CN slot entropy, and the number of alternatives in the CN slot. The raw language model and acoustic scores are also available. In addition, it includes a frame-based confidence that is computed from only the acoustic model, and a frame-based confidence that is computed from only the language model score. Frame-based confidences are calculated from the lattices according to (Wessel et al., 1998); the CN-algorithm is an extension of (Xue and Zhao, 2005).

The next class of features describes *durational* aspects of the top hypothesis for each system, including: character length, frame duration, frames per character, and if the word is the empty or null word. A feature that normalizes the frames per character by the average over a window of ten words is also generated. Here we use characters as a proxy for phones, because phone information is not available from all component systems.

We also identify the system dependent *top error* words for the development set, as well as the words that occur to the left and right of the system errors. We encode this information by indicating if a system word is on the list of top ten errors or the top one hundred list, and likewise if the left or right system context word is found in their corresponding lists.

In order to provide *comparisons* across systems, we compute the character distance (the cost of aligning the words at the character level) between the system words

and provide that as a feature. In addition, we include the confidence of a system word as computed by the frame-wise posteriors of each of the other systems. This allows each of the other systems to 'score' the hypothesis of a system in question. These cross-system confidences could also act as an indicator for when one system's hypothesis is an OOV-word for another system. We also compute the standard, confidence-based ROVER hypothesis at each slot, and indicate whether or not a system agrees with ROVER's decision.

The last set of features is computed relative to the combined *min-fWER* decoding. A confidence for each system word is calculated from the combined frame-wise posteriors of all component systems. The final feature indicates whether each system word agrees with the combined systems' min-fWER hypothesis.

3.4 Classifier

After producing a set of features to characterize the systems, we train a classifier with these features that will decide which system will propose the final hypothesis at each slot in the multiple alignment. The target classes include one for each system and a null class (which is selected when none of the system outputs are chosen, i.e. a system insertion).

The training data begins with the multiple alignment of the hypothesis systems, which is then aligned to the reference words. The learning target for each slot is the set of systems which match the reference word, or the null class if no systems match the reference word. Only slots where there is disagreement between the systems' 1-best hypotheses are included in training and testing.

The classifier for our work is Boostexter (Schapire and Singer, 2000) using real Adaboost.MH with logistic loss (which outperformed exponential loss in preliminary experiments). Boostexter trains a series of weak classifiers (tree stumps), while also updating the weights of each training sample such that examples that are harder to classify receive more weight. The weak classifiers are then combined with the weights learned in training to predict the most likely class in testing. The main dimensions for model tuning are feature selection and number of iterations, which are selected on the development set as described in the next section.

4 Experiments

We first perform experiments using cross-validation on the development set to determine the impact of different feature classes, and to select the optimal number of iterations for Boostexter training. We then apply the models to the evaluation set.

4.1 Experimental setup

In our experiments we combine lattice sets for the English task of the TC-STAR 2006 Evaluation Campaign from four sites. The TC-STAR project partners kindly provided RWTH their development and evaluation lattices. Systems and lattice sets are described in (Hoffmeister et al., 2007).

Table 1 summarizes the best results achieved on the single lattice sets. The latter columns show the results of CN and min-fWER based posterior decoding (Mangu et al., 2000; Wessel et al., 2001).

	Viterbi		min-fWER		CN	
	dev	eval	dev	eval	dev	eval
1	10.5	9.0	10.3	8.6	10.4	8.6
2	11.4	9.0	11.4	9.5	11.6	9.1
3	12.8	10.4	12.5	10.4	12.6	10.2
4	13.9	11.9	13.9	11.8	13.9	11.8

Table 1: *WER[%] results for single systems.*

4.2 Feature analysis on development data

We evaluate the various feature classes from Section 3.3 on the development set with a cross validation testing strategy. The results in Tables 2 and 3 are generated with ten-fold cross validation, which maintains a clean separation of training and testing data. The total number of training samples (alignment slots where there is system disagreement) is about 3,700 for the 2 system case, 5,500 for the 3 system case, and 6,800 for the 4 system case.

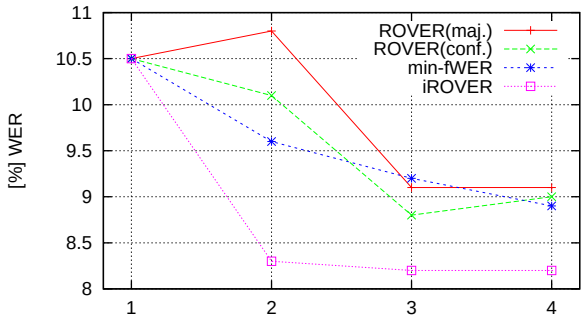
The WER results for different feature conditions on the development set are presented in Table 2. The typical ROVER with word confidences is provided in the first row for comparison, and the remainder of the rows contain the results for various configurations of features that are made available to the classifier.

The *basic* features are just those that encode the same information as ROVER, but the classifier is still able to learn better decisions than ROVER with only these features. Each of the following rows provides the results for adding a single feature class to the *basic* features, so that the impact of each type can be evaluated.

The last two rows contain combinations of feature classes. First, the best three classes are added, and then all features. Using just the best three classes achieves almost the best results, but a small improvement is gained when all features are added. The number of iterations in training is also optimized on the development set by selecting the number with the lowest average classification error across the ten splits of the training data.

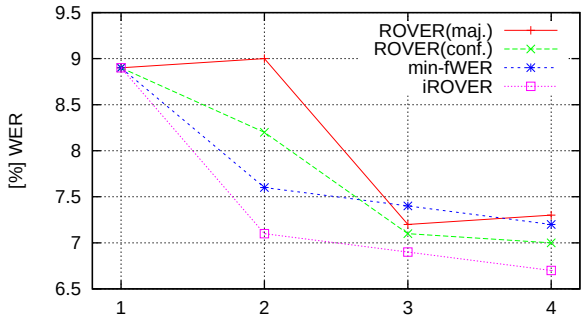
Features	2 System	3 System	4 System
ROVER	10.2%	8.8%	9.0%
basic	9.4%	8.6%	8.5%
+confidences	9.3%	8.7%	8.4%
+durational	9.2%	8.6%	8.4%
+top error	9.0%	8.5%	8.4%
+comparisons	8.9%	8.6%	8.4%
+min-fWER	8.5%	8.5%	8.4%
+top+cmp+fWER	8.3%	8.3%	8.2%
all features	8.3%	8.2%	8.2%

Table 2: WER results for development data with different feature classes.



	2 System	3 System	4 System
ROVER (maj.)	10.8%	9.1%	9.1%
ROVER (conf.)	10.1%	8.8%	9.0%
min-fWER	9.6%	9.2 %	8.9 %
*i*ROVER	8.3%	8.2%	8.2%
oracle	6.5%	5.4%	4.7%

Table 3: *WER[%] results for development data with manual segmentation, and using cross-validation for iROVER.*



	2 System	3 System	4 System
ROVER(maj.)	9.0%	7.2%	7.3%
ROVER(conf.)	8.2%	7.1%	7.0%
min-fWER	7.6 %	7.4 %	7.2 %
*i*ROVER	7.1%	6.9%	6.7%
oracle	5.2%	4.1%	3.6%

Table 4: *WER[%] results for evaluation data.*

4.3 Results on evaluation data

After analyzing the features and selecting the optimal number of training iterations on the development data, we train a final model on the full development set and then apply it to the evaluation set. In all cases our classifier achieves a lower WER than ROVER (statistically significant by NIST matched pairs test). Table 3 and Table 4 present a comparison of the ROVER with majority voting, confidence voting, frame-based combination, and our improved ROVER (*i*ROVER).

5 Conclusions

In summary, we develop *i*ROVER, a method for system combination that outperforms ROVER consistently across varying numbers of component systems. The relative improvement compared to ROVER is especially large for the case of combining two systems (14.5% on the evaluation set). The relative improvements are larger than any we know of to date, and the four system case achieves the best published result on the TC-STAR English evaluation set. The classifier requires relatively little training data and utilizes features easily available from system lattices.

Future work will investigate additional classifiers, classifier combination, and expanded training data. We are also interested in applying a language model to decode an alignment network that has been scored with our classifier.

References

G. Evermann and P. Woodland. 2000. Posterior probability decoding, confi dence estimation and system combination. In *NIST Speech Transcription Workshop*.

J.G. Fiscus. 1997. A post-processing system to yield reduced word error rates: Recognizer Output Voting Error Reduction (ROVER). In *Proc. ASRU*.

B. Hoffmeister, T. Klein, R. Schlüter, and H. Ney. 2006. Frame based system combination and a comparison with weighted ROVER and CNC. In *Proc. ICSLP*.

B. Hoffmeister, D. Hillard, S. Hahn, R. Schülter, M. Ostendorf, and H. Ney. 2007. Cross-site and intra-site ASR system combination: Comparisons on lattice and 1-best methods. In *Proc. ICASSP*.

L. Mangu, E. Brill, and A. Stolcke. 2000. Finding consensus in speech recognition: word error minimization and other applications of confusion networks. *Computer Speech and Language*, 14:373–400.

R. E. Schapire and Y. Singer. 2000. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168.

A. Stolcke, H. Bratt, J. Butzberger, H. Franco, V. Gadde, M. Plauche, C. Richey, E. Shriberg, K. Sonmez, J. Zheng, and F. Weng. 2000. The SRI March 2000 Hub-5 conversational speech transcription system. In *NIST Speech Transcription Workshop*.

F. Wessel, K. Macherey, and R. Schlüter. 1998. Using word probabilities as confi dence measures. In *Proc. ICASSP*.

F. Wessel, R. Schlüter, and H. Ney. 2001. Explicit word error minimization using word hypothesis posterior probabilities. In *Proc. ICASSP*, volume 1.

Jian Xue and Yunxin Zhao. 2005. Improved confusion network algorithm and shortest path search from word lattice. In *Proc. ICASSP*.

R. Zhang and A. Rudnicky. 2006. Investigations of issues for using multiple acoustic models to improve continuous speech recognition. In *Proc. ICSLP*.

Clustered Sub-matrix Singular Value Decomposition

Fang Huang
School of Computing
Robert Gordon University
Aberdeen, AB25 1HG, UK
f.huang@rgu.ac.uk

Yorick Wilks
Department of Computer Science
University of Sheffield
Sheffield, S1 4DP, UK
y.wilks@dcs.shef.ac.uk

Abstract

This paper presents an alternative algorithm based on the singular value decomposition (SVD) that creates vector representation for linguistic units with reduced dimensionality. The work was motivated by an application aimed to represent text segments for further processing in a multi-document summarization system. The algorithm tries to compensate for SVD's bias towards dominant-topic documents. Our experiments on measuring document similarities have shown that the algorithm achieves higher average precision with lower number of dimensions than the baseline algorithms - the SVD and the vector space model.

1 Introduction

We present, in this paper, an alternative algorithm called Clustered Sub-matrix Singular Value Decomposition(CSSVD) algorithm, which applied clustering techniques before basis vector calculation in SVD (Golub and Loan, 1996). The work was motivated by an application aimed to provide vector representation for terms and text segments in a document collection. These vector representations were then used for further preprocessing in a multi-document summarization system.

The SVD is an orthogonal decomposition technique closely related to eigenvector decomposition and factor analysis. It is commonly used in information retrieval as well as language analysis applications. In SVD, a real m-by-n matrix A is decomposed into three matrices, $A = U \sum V^T$. $\sum$ is an m-by-n matrix such that the singular value σ_i=$\sum_{ii}$ is the square root of the ith largest eigenvalue of AA^T, and $\sum_{ij} = 0$ for $i \neq j$. Columns of orthogonal matrices U and V define the orthonormal eigenvectors associated with eigenvalues of AA^T and A^TA, respectively. Zeroing out all but the k, $k < rank(A)$, largest singular values yields $A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$, which is the closest rank-k matrix to A. Let A be a term-document matrix. Applications such as latent semantic indexing (Deerwester et al., 1990) apply the rank-k approximation A_k to the original matrix A, which corresponds to projecting A onto the k-dimension subspace spanned by $u_1, u_2, ..., u_k$. Because $k \ll m$, in this k-dimension space, minor terms are ignored, so that terms are not independent as they are in the traditional vector space model. This allows semantically related documents to be related to each other even though they may not share terms.

However, SVD tends to wipe out outlier (minority-class) documents as well as minor terms (Ando, 2000). Consequently, topics underlying outlier documents tend to be lost. In applications such as multi-document summarization, a set of related documents are used as the information source. Typically, the documents describe one broad topic from several different view points or sub-topics. It is important for each of the sub-topics underlying the document collection to be represented well.

Based on the above consideration, we propose the CSSVD algorithm with the intention of compensat-

Proceedings of NAACL HLT 2007, Companion Volume, pages 69–72,
Rochester, NY, April 2007.

ing for SVD's tendency to wipe out minor topics. The basic idea is to group the documents into a set of clusters using clustering algorithms. The SVD is then applied on each of the document clusters. The algorithm thus selects basis vectors by treating equally each of the topics. Our experiments on measuring document similarities have shown that the algorithm achieves higher average precision with lower number of dimensions than the SVD.

2 the Algorithm

The input to the CSSVD algorithm is an m×n term-document matrix A. Documents in matrix A are grouped into a set of document clusters. Here, we adopt single-link algorithm to develop the initial clusters, then use K-means method to refine the clusters. After clustering, columns in matrix A are partitioned and regrouped into a set of sub-matrices A_1,A_2,...,A_q. Each of these matrices represents a document cluster. Assume A_i, $1 \leq i \leq q$, is an $m \times n_i$ matrix, these sub-matrices are ranked in decreasing order of their sizes, i.e., $n_1 \geq n_2 \geq ... \geq n_q$, then $n_1 + n_2 + ... + n_q = n$.

The algorithm computes basis vectors as follows: the first basis vector u_1 is computed from A_1, i.e., the first left singular vector of A_1 is selected. In order to ensure that the basis vectors are orthogonal, singular vectors are actually computed on residual matrices. R_{ij}, the residual matrix of A_i after the selection of basis vectors u_1, u_2,..., u_j, is defined as

$$R_{ij} = \begin{cases} A_i & j = 0 \\ A_i - proj(A_{ij}) & otherwise \end{cases}$$

where, $proj(A_{ij})$ is the orthogonal projection of the document vectors in A_i onto the span of u_1,u_2,...,u_j, i.e.,

$$proj(A_{ij}) = \sum_{k=1}^{j} u_k u_k^T A_i$$

the residual matrix of A_i describes how much the document vectors in A_i are excluded from the proposed basis vectors u_1, u_2,..., u_j. For the first basis vector computation, residual matrices are initialized as original sub-matrices. The computation of the residual matrix makes the remaining vectors perpendicular to the previous basis vectors, thus ensures that the basis vectors are orthogonal, as the eigenvector computed next is a linear combination of the remaining vectors.

After calculating a basis vector, the algorithm judges whether the sub-matrices have been well represented by the derived basis vectors. The residual ratio was defined as a criterion for this judgement,

$$rr_{ij} = \frac{||R_{ij}||_F^2}{n_i \times (k_i + 1)}$$

where R_{ij} is the residual matrix of A_i after j basis vectors have been selected[1]; n_i is the number of the documents in matrix A_i; k_i is the number of singular vectors that have been selected from matrix A_i. Residual ratios of each sub-matrix are calculated. The sub-matrix with the largest residual ratio is assumed to be the one that contains the most information that has not been represented by the previous chosen basis vectors. The first left singular vector of this sub-matrix is computed and selected as the next basis vector. As described above, the computation of a basis vector uses the corresponding residual matrix. Once a basis vector is selected, its influence from each sub-matrix is subtracted. The procedure is repeated until an expected number of basis vectors have been chosen. The pseudo-code of the algorithm for semantic space construction is shown as follows:

1. Partition A into matrices A_1,...,A_q corresponding to document clusters, where A_i , $1 \leq i \leq q$, is an $m \times n_i$ ($n_1 \geq n_2 \geq ... \geq n_q$) matrix.
2. For i=1,2,...,q $\{R_i = A_i$; k[i]=0;$\}$
3. j=1; r=1;
4. u_r= the first unit eigenvector of $R_j R_j^T$;
5. For i=1,2,...,q $R_i = R_i - u_r u_r^T R_i$;
6. k[r]=k[r]+1; r=r+1;
7. For i=1,2,...,q $rr_i = \frac{||R_i||_F^2}{(n_i \times (k[i]+1))}$;
8. j=t if $rr_t > rr_p$ for p=1,2,...,q and $p \neq t$;
9. If $rr_j \leq threshold$ then stop else goto step 4.

For the single-link algorithm used in the CSSVD, we use a threshold 0.2 and cosine measure to calculate the similarity between two clusters in our experiments. The performance of the CSSVD is also relative to the number of dimensions of the created

[1] $||A||_F = \sqrt{\sum_{i,j} A_{ij}^2}$

subspace. As described above, the algorithm uses the residual ratio as a stopping criterion for the basis vector computation. In each iteration, after a basis vector is created, the residual ratio is compared to a threshold. Once the residual ratio of each sub-matrix fell below a certain threshold, the process of basis-vector selection is finished. In our experiments, the threshold was trained on corpus.

After all the k basis vectors are chosen, a term-document vector d_i can be converted to d_i^k, a vector in the k-dimensional space, by multiplying the matrix of basis vectors following the standard method of orthogonal transformation,i.e., $d_i^k = [u_1, u_2, ..., u_k]^T d_i$.

3 Evaluation

3.1 Experimental Setup

For the evaluation of the algorithm, 38 topics from the Text REtrieval Conference (TREC) collections were used in our experiments. These topics include foreign minorities, behavioral genetics, steel production, etc. We deleted documents relevant to more than one topic so that each document is related only to one topic. The total number of documents used was 2962. These documents were split into two disjoint groups, called 'pool 1' and 'pool 2'. The number of documents in 'pool 1' and 'pool 2' were 1453 and 1509, respectively. Each of the two groups used 19 topics.

We generated training and testing data by simulating the result obtained by a query search. This simulation is further simplified by selecting documents containing same keywords from each document group. Thirty document sets were generated from each of the two document groups, i.e. 60 document sets in total. The number of documents for each set ranges from 51 to 582 with an average of 128; the number of topics ranges from 5 to 19 with an average of 12. Due to the limited number of the document sets we created, these sets were used both for training and evaluation. For the evaluation of the documents sets from 'pool 1', 'pool 2' was used for training, and vice versa.

To construct the original term-document matrix, the following operations were performed on each of the documents: 1) filtering out all non-text tags in the documents; 2) converting all the characters into lower case; 3) removing stop words - a stoplist containing 319 words was used; and 4) term indexing - the tf.idf scheme was used to calculate a term's weight in a document. Finally, a document set is represented as a matrix $A = [a_{ij}]$, where a_{ij} denotes the normalized weight assigned to term i in document j.

3.2 Evaluation Measures

Our algorithm was motivated by a multi-document summarization application which is mainly based on measuring the similarities and differences among text segments. Therefore, the basic requisite is to accurately measure similarities among texts. Based on this consideration, we used the CSSVD algorithm to create the document vectors in a reduced space for each of the document sets; cosine similarities among these document vectors were computed; and the results were then compared with the TREC relevance judgments. As each of the TREC documents we used has one specific topic. Assume that similarity should be higher for any document pair relevant to the same topic than for any pair relevant to different topics. The algorithm's accuracy for measuring the similarities among documents was evaluated using average precision taken at various recall levels (Harman, 1995). Let p_i denote the document pair that has the ith largest similarity value among all pairs of documents in the document set. The precision for an intra-topic pair p_k is calculated by

$$precision(p_k) = \frac{number\ of\ p_j\ where\ j \leq k}{k}$$

where p_j is an intra-topic pair. The average of the precision values over all intra-topic pairs is computed as the average precision.

3.3 Results

The algorithms are evaluated by the average precision over 60 document sets. In order to make a comparison, two baseline algorithms besides CSSVD are evaluated. One is the vector space model (VSM) without dimension reduction. The other is SVD taking the left singular vectors as the basis vectors.

To treat the selection of dimensions as a separate issue, we first evaluate the algorithms in terms of the best average precision. The 'best average precision' means the best over all the possible numbers

of dimensions. The second row of Table 1 shows the best average precision of our algorithm, VSM, and SVD. The best average precision on average over 60 document sets of CSSVD is 69.6%, which is 11.5% higher than VSM and 6.1% higher than SVD.

measure	VSM	SVD	CSSVD
best average precision (%)	58.1	63.5	69.6
average DR (%)	N/A	54.4	32.1
average precision (%)	58.1	59.5	66.8

Table 1: the algorithm performance

In the experiments, we observed that the CSSVD algorithm obtained its best performance with the number of dimensions lower than that of SVD. The Dimensional Ratio (DR) is defined as the number of dimensions of the derived sub-space compared with the dimension number of the original space, i.e.,

$$DR = \frac{\#\ of\ dimensions\ in\ derived\ space}{\#\ of\ dimensions\ in\ original\ space}$$

The average dimensional ratio is calculated over all the 60 document sets. As the algorithms' computational efficiency is dependent on the number of dimensions computed, our interest is in getting good performance with an average dimensional ratio as low as possible. The third row of Table 1 shows the average dimensional ratio that yielded the best average precision. The average dimensional ratio that CSSVD yielded the best average precision is 32.1%, which is 22.3% lower than that of SVD. Thus, our algorithm has the advantage of being computationally inexpensive, assuming that we can find the optimal number of dimensions.

The bottom row of Table 1 shows the average precision of the algorithms. The threshold used in CSSVD algorithm was trained on corpus. Let p be the threshold on residual ratio that yielded the best average precision on the training data. The value of p is then used as the threshold on the evaluation data. For the SVD algorithm, the average dimensional ratio that yielded the best average precision on training data was used as the dimensional ratio to determine the subspace dimensionality in evaluation. The performance shown here are the average of average precision over 60 document sets. Again, the CSSVD achieves the best performance, which is 7.3% higher than the performance of SVD and 8.7% higher than VSM.

4 Conclusion

We have presented an alternative algorithm, the CSSVD, that creates vector representation for linguistic units with reduced dimensionality. The algorithm aims to compensate for SVD's bias towards dominant-topic documents by grouping documents into clusters and selecting basis vectors from each of the clusters. It introduces a threshold on the residual ratio of clusters as a stopping criterion of basis vector selection. It thus treats each topic underlying the document collection equally while focuses on the dominant documents in each topic. The preliminary experiments on measuring document similarities have shown that the CSSVD achieves higher average precision with lower number of dimensions than the baseline algorithms.

Motivated by a multi-document summarization application, the CSSVD algorithm's emphasis on topics and dominant information within each topic meets the general demand of summarization. We expect that the algorithm fits the task of summarization better than SVD. Our future work will focus on more thorough evaluation of the algorithm and integrating it into a summarization system.

5 Acknowledgments

We would like to thank Mark Sanderson, Horacio Saggion, and Robert Gaizauskas for helpful comments at the beginning of this research.

References

Ando R.K. 2000 Latent Sementic Space: Iterative Scaling Improves Precision of Inter-document Similarity Measurement. *Proceedings of ACM SIGIR 2000*, Athens, Greece.

Deerwester S., Dumais S., Furnas G., and Landauer T. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41:391-407.

Golub G. and Loan C.V. 1996. *Matrix Computations*. Johns-Hopkins University Press, Maryland, US.

Harman D.K. 1983. Overview of the second Text Retrieval Conference (TREC-2). *Information Processing Management*, 31(3):271-289.

Implicitly Supervised Language Model Adaptation for Meeting Transcription

David Huggins-Daines
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
dhuggins@cs.cmu.edu

Alexander I. Rudnicky
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
air@cs.cmu.edu

Abstract

We describe the use of meeting metadata, acquired using a computerized meeting organization and note-taking system, to improve automatic transcription of meetings. By applying a two-step language model adaptation process based on notes and agenda items, we were able to reduce perplexity by 9% and word error rate by 4% relative on a set of ten meetings recorded in-house. This approach can be used to leverage other types of metadata.

1 Introduction

Automatic transcription of multi-party conversations such as meetings is one of the most difficult tasks in automatic speech recognition. In (Morgan et al., 2003) it is described as an "ASR-complete" problem, one that presents unique challenges for every component of a speech recognition system.

Though much of the literature on meeting transcription has focused on the unique acoustic modeling and segmentation problems incurred by meeting transcription, language modeling for meetings is an interesting problem as well. Though meeting speech is spontaneous in nature, the vocabulary and phrasing in meetings can be very specialized and often highly technical. Speaking style can vary greatly between speakers, and the discourse structure of multi-party interaction gives rise to cross-speaker effects that are difficult to model with standard N-gram models (Ji and Bilmes, 2004).

Speech in meetings has one crucial advantage over many other transcription tasks, namely that it does not occur in isolation. Meetings are scheduled and discussed in advance, often via e-mail. People take notes and create agendas for meetings, and often read directly from electronic presentation materials. The structure of meetings can be exploited - topics can be segmented both temporally and across speakers, and these shifting topics can be modeled as sub-languages.

We examine the effect of leveraging one particular type of external information, namely the written agendas and meeting minutes, and we demonstrate that, by using off-line language model adaptation techniques, these can significantly ($p < 0.01$) improve language modeling and speech recognition accuracy. The language in the notes and agendas is very similar to that used by the speakers, hence we consider this to be a form of semi-supervised or *implicitly supervised* adaptation.

2 Corpus

The SmartNotes system, described in (Banerjee and Rudnicky, 2007) is a collaborative platform for meeting organization, recording, and note-taking. As part of our research into meeting segmentation and recognition, we have collected a series of 10 unscripted meetings using SmartNotes. These meetings themselves are approximately 30 minutes in length (ranging from 1745 to 7208 words) with three regular participants, and consist of discussions and reporting on our ongoing research. The meetings are structured around the agendas and action items constructed through the SmartNotes interface. The

Proceedings of NAACL HLT 2007, Companion Volume, pages 73–76,
Rochester, NY, April 2007.

agenda itself is largely constant from meeting to meeting, while each meeting typically reviews discusses the previous week's action items. Each participant is equipped with a laptop computer and an individual headset microphone.

Each meeting was manually transcribed and segmented for training and testing purposes. The transcription includes speaker identification and timing information. As part of the meeting, participants are encouraged to take notes and define action items. These are automatically collected on a server along with timestamp information. In (Banerjee and Rudnicky, 2007), it was shown that timestamped text of this kind is useful for topic segmentation of meetings. In this work, we have not attempted to take advantage of the timing information, nor have we attempted to perform any topic segmentation. Given the small quantity of text available from the notes, we feel that the type of static language model adaptation presented here is most feasible when done at the entire meeting level. A cache language model (Kuhn and Mori, 1990) may be able to capture the (informally attested) locality effects between notes and speech.

Since the notes are naturalistic text, often containing shorthand, abbreviations, numbers, punctuation, and so forth, we preprocess them by running them through the text-normalization component of the Festival[1] speech synthesis system and extracting the resulting string of individual words. This yielded an average of 252 words of adaptation data for each of the 10 meetings.

3 System Description

Unless otherwise noted, all language models evaluated here are trigram models using Katz smoothing (Katz, 1987) and Good-Turing discounting. Linear interpolation of multiple source models was performed by maximizing the likelihood over a held-out set of adaptation data.

For automatic transcription, our acoustic models consist of 5000 tied triphone states (senones), each using a 64-component Gaussian mixture model with diagonal covariance matrices. The input features consist of 13-dimensional MFCC features, delta, and delta-delta coefficients. These models are trained on approximately 370 hours of speech data, consisting of the ICSI meeting corpus (Morgan et al., 2003), the HUB-4 Broadcast News corpus, the NIST pilot meeting corpus, the WSJ CSR-0 and CSR-1 corpora,[2] the CMU Arctic TTS corpora (Kominek and Black, 2004), and a corpus of 32 hours of meetings previously recorded by our group in 2004 and 2005.

Corpus	# Words	Perplexity
Fisher English	19902585	178.41
Switchboard-I	2781951	215.52
ICSI (75 Meetings)	710115	134.94
Regular Meetings	266043	111.76
Switchboard Cellular	253977	280.81
CallHome English	211377	272.19
NIST Meetings	136932	199.40
CMU (ISL Meetings)	107235	292.86
Scenario Meetings	36694	306.43

Table 1: Source Corpora for Language Model

Our baseline language model is based on a linear interpolation of source language models built from conversational and meeting speech corpora, using a held-out set of previously recorded "scenario" meetings. These meetings are unscripted, but have a fixed topic and structure, which is a fictitious scenario involving the hiring of new researchers. The source language models contain a total of 24 million words from nine different corpora, as detailed in Table 1. The "Regular Meetings" and "Scenario Meetings" were collected in-house and consist of the same 32 hours of meetings mentioned above, along with the remainder of the scenario meetings. We used a vocabulary of 20795 words, consisting of all words from the locally recorded, ICSI, and NIST meetings, combined with the Switchboard-I vocabulary (with the exception of words occurring less than 3 times). The Switchboard and Fisher models were pruned by dropping singleton trigrams.

4 Interpolation and Vocabulary Closure

We created one adapted language model for each meeting using a two-step process. First, the source language models were re-combined using linear interpolation to minimize perplexity on the set of notes

[1] http://www.festvox.org/

[2] All corpora are available through http://ldc.upenn.edu/

Meeting	Baseline	Interpolated	Closure
04/17	90.05	85.96	84.41
04/24	90.16	85.54	81.88
05/02	94.27	89.24	89.19
05/12	110.95	101.68	87.13
05/18	85.78	81.50	78.04
05/23	97.51	93.07	94.39
06/02	109.70	104.49	101.90
06/12	96.80	92.88	91.05
06/16	93.93	87.71	79.17
06/20	97.19	93.88	92.48
Mean	96.57	91.59 (-5.04)	87.96 (-8.7)
S.D.	8.61	7.21 (1.69)	7.40 (6.2)
p	n/a	< 0.01	< 0.01

Table 2: Adaptation Results: Perplexity

Meeting	Baseline	Interpolated	Closure
04/17	45.22	44.37	43.34
04/24	47.35	46.43	45.25
05/02	47.20	47.20	46.28
05/12	49.74	48.02	46.07
05/18	45.29	44.63	43.44
05/23	43.68	43.00	42.80
06/02	48.66	48.29	47.85
06/12	45.68	45.90	45.28
06/16	45.98	45.45	44.29
06/20	47.03	46.73	46.68
Mean	46.59	46.0 (-0.58)	45.13 (-1.46)
S.D.	1.78	1.68 (0.54)	1.64 (1.0)
p	n/a	< 0.01	< 0.01

Table 3: Adaptation Results: Word Error

for each meeting. Next, the vocabulary was expanded using the notes. In order to accomplish this, a trigram language model was trained from the notes themselves and interpolated with the output of the previous step using a small, fixed interpolation weight $\lambda = 0.1$. It should be noted that this also has the effect of slightly boosting the probabilities of the N-grams that appear in the notes. We felt this was useful because, though these probabilities are not reliably estimated, it is likely that people will use many of the same N-grams in the notes as in their meeting speech, particularly in the case of numbers and acronyms. The results of interpolation and N-gram closure are shown in Table 2 in terms of test-set perplexity, and in Table 3 in terms of word error rate. Using a paired t-test over the 10 meetings, the improvements in perplexity and accuracy are highly significant ($p < 0.01$).

5 Topic Clustering and Dimensionality Reduction

In examining the interpolation component of the adaptation method described above, we noticed that the in-house meetings and the ICSI meetings consistently took on the largest interpolation weights. This is not surprising since both of these corpora are similar to the test meetings. However, all of the source corpora cover potentially relevant topics, and by interpolating the corpora as single units, we have no way to control the weights given to individual topics within them. Also, people may use different, but related, words in writing and speaking to describe the same topic, but we are unable to capture these semantic associations between the notes and speech.

To investigate these issues, we conducted several brief experiments using a reduced training corpus consisting of 69 ICSI meetings. We converted these to a vector-space representation using *tf.idf* scores and used a *deterministic annealing* algorithm (Rose, 1998) to create hard clusters of meetings, from each of which we trained a source model for linear interpolation. We compared these clusters to random uniform partitions of the corpus. The interpolation weights were trained on the notes, and the models were tested on the meeting transcripts. Out-of-vocabulary words were not removed from the perplexity calculation. The results (mean and standard deviation over 10 meetings) are shown in Table 4. For numbers of clusters between 2 and 42, the annealing-based clusters significantly outperform the random partition. The perplexity with 42 clusters is also significantly lower ($p < 0.01$) than the perplexity (256.5 ± 21.5) obtained by training a separate source model for each meeting.

To address the second issue of vocabulary mismatches between notes and speech, we applied *probabilistic latent semantic analysis* (Hofmann, 1999) to the corpus, and used this to "expand" the vocabulary of the notes. We trained a 32-factor PLSA model on the content words (we used a simple

# of Clusters	Random	Annealed
2	546.5 ± 107.4	514.1 ± 97.9
4	462.2 ± 86.3	426.2 ± 73.9
8	397.7 ± 67.1	356.1 ± 54.9
42	281.6 ± 31.5	253.7 ± 22.9

Table 4: Topic Clustering Results: Perplexity

Meeting	Baseline	PLSA	"Boosted"
04/17	105.49	104.59	104.87
04/24	98.97	97.58	97.80
05/02	105.61	104.15	104.48
05/12	122.37	116.73	118.04
05/18	98.55	94.92	95.18
05/23	111.28	107.84	108.03
06/02	125.31	121.49	121.64
06/12	109.31	106.38	106.55
06/16	106.86	103.27	104.28
06/20	117.46	113.76	114.18
Mean	110.12	107.07	107.50
S.D.	8.64	7.84	7.93
p	n/a	< 0.01	< 0.01

Table 5: PLSA Results: Perplexity

entropy-based pruning to identify these "content words") from the ICSI meeting vocabulary. To adapt the language model, we used the "folding-in" procedure described in (Hofmann, 1999), running an iteration of EM over the notes to obtain an adapted unigram distribution. We then simply updated the unigram probabilities in the language model with these new values and renormalized. While the results, shown in Table 5, show a statistically significant improvement in perplexity, this adaptation method is is problematic, as it increases the probability mass given to all the words in the PLSA model. In subsequent results, also shown in Table 5, we found that simply extracting these words from the original unigram distribution and boosting their probabilities by the equivalent amount also reduces perplexity by nearly as much (though the difference from the PLSA model is statistically significant, $p = 0.004$).

6 Conclusions

We have shown that notes collected automatically from participants in a structured meeting situation can be effectively used to improve language modeling for automatic meeting transcription. Furthermore, we have obtained some encouraging results in applying source clustering and dimensionality reduction to make more effective use of this data. In future work, we plan to exploit other sources of metadata such as e-mails, as well as the structure of the meetings themselves.

7 Acknowledgements

This research was supported by DARPA grant NG CH-D-03-0010. The content of the information in this publication does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred.

References

S. Banerjee and A. I. Rudnicky. 2007. Segmenting meetings into agenda items by extracting implicit supervision from human note-taking. In *Proceedings of the 2007 International Conference on Intelligent User Interfaces*, January.

Thomas Hofmann. 1999. Probabilistic latent semantic analysis. In *Proceedings of UAI'99*, Stockholm.

G. Ji and J. Bilmes. 2004. Multi-speaker language modeling. In *Proceedings of HLT-NAACL*.

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

J. Kominek and A. Black. 2004. The CMU Arctic speech databases. In *5th ISCA Speech Synthesis Workshop*, Pittsburgh.

R. Kuhn and R. De Mori. 1990. A cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 570–583.

N. Morgan, D. Baron, S. Bhagat, R. Dhillon H. Carvey, J. Edwards, D. Gelbart, A. Janin, A. Krupski, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. 2003. Meetings about meetings: research at ICSI on speech in multiparty conversation. In *Proceedings of ICASSP*, Hong Kong, April.

K. Rose. 1998. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. In *Proceedings of the IEEE*, pages 2210–2239.

ILR-Based MT Comprehension Test with Multi-Level Questions

Douglas Jones, Martha Herzog, Hussny Ibrahim, Arvind Jairam, Wade Shen,
Edward Gibson and Michael Emonts

MIT Lincoln Laboratory
Lexington, MA 02420
{DAJ,Arvind,SWade}@LL.MIT.EDU
MHerzog2005@comcast.net

DLI Foreign Language Center
Monterey, CA 93944
{Hussny.Ibrahim,Michael.Emonts}
@monterey.army.mil

MIT Brain and Cognitive
Sciences Department
Cambridge MA, 02139
EGibson@MIT.EDU

Abstract

We present results from a new Interagency Language Roundtable (ILR) based comprehension test. This new test design presents questions at multiple ILR difficulty levels within each document. We incorporated Arabic machine translation (MT) output from three independent research sites, arbitrarily merging these materials into one MT condition. We contrast the MT condition, for both text and audio data types, with high quality human reference Gold Standard (GS) translations. Overall, subjects achieved 95% comprehension for GS and 74% for MT, across 4 genres and 3 difficulty levels. Surprisingly, comprehension rates do not correlate highly with translation error rates, suggesting that we are measuring an additional dimension of MT quality. We observed that it takes 15% more time overall to read MT than GS.

1 Introduction

The official Defense Language Proficiency Test (DLPT) is constructed according to rigorous and well-established principles that have been developed to measure the foreign language proficiency of human language learners in U.S. Department of Defense settings. In 2004, a variant of that test type was constructed, following the general DLPT design principles, but modified to measure the quality of machine translation. This test, known as the DLPTstar (Jones et al, 2005), was based on authentic Arabic materials at ILR text difficulty levels 1, 2, and 3, accompanied by constructed-response questions at matching levels. The ILR level descriptors, used throughout the U.S. government, can be found at the website cited in the list of references. The text documents were presented in two conditions in English translation: (1) professionally translated into English, and (2) machine translated with state-of-the art MT systems, often quite garbled. Results showed that native readers of English could generally pass the Levels 1 and 2 questions on the test, but not those at Level 3. Also, Level 1 comprehension was less than expected, given the low level of the original material. It was not known whether the weak Level 1 performance was due to systematic deficits in MT performance at Level 1, or whether the materials were simply mismatched to the MT capabilities.

In this paper, we present a new variant of the test, using materials specifically created to test the capabilities of the MT systems. To guarantee that the MT systems were up to the task of processing the documents, we used the DARPA GALE 2006 evaluation data sets, against which several research sites were testing MT algorithms. We arbitrarily merged the MT output from three sites. The ILR difficulty of the documents ranged from Level 2 to Level 3, but the test did not contain any true Level 1 documents. To compensate for this lack, we constructed questions about Level 1 elements (e.g., personal and place names) in Level 2 and 3 documents. A standard DLPT would have more variation at Level 1.

2 Related and Previous Work

Earlier work in MT evaluation incorporated an informativeness measure, based on comprehension test answers, in addition to fluency, a measure of output readability without reference to a gold standard, and adequacy, a measure of accuracy with reference to a gold standard translation (White and O'Connell, 1994). Later MT evaluation found fluency and adequacy to correlate well enough with automatic measures (BLEU), and since comprehension tests are relatively more expensive to create, the informativeness test was not used in later

Proceedings of NAACL HLT 2007, Companion Volume, pages 77–80,
Rochester, NY, April 2007.

MT evaluations, such as the ones performed by NIST from 2001-2006. In other work, task-based evaluation has been used for MT evaluation (Voss and Tate, 2006), which measures human performance on exhaustively extracting 'who', 'when', and 'where' type elements in MT output. The DLPT-star also uses this type of factual question, particularly for Level 2 documents, but not exhaustively. Instead, the test focuses on text elements most characteristic of the levels as defined in the ILR scale. At Level 3, for example, questions may concern abstract concepts or hypotheses found in the documents. Applying the ILR construct provides Defense Department decision makers with test scores that are readily interpretable.

3 Test Construction and Administration

In this paper, we present a new test, based entirely on the DARPA GALE 2006 evaluation data, selecting approximately half of the material for our test. We selected twenty-four test documents, with balanced coverage across four genres: newswire, newsgroups, broadcast news and talk radio. Our target was to have at least 2500 words for each genre, which we exceeded slightly with approximately 12,200 words in total for the test. We began with a random selection of documents and adjusted it for better topic coverage. We constructed an exhaustive set of questions for each document, approximately 200 questions in total. The questions ranged in ILR difficulty, from "0+, 1,1+, 2, 2+ and 3, with Levels 0+, 1 and 1+ combined to a pseudo-level we called L1~, providing four levels of difficulty to be measured. We divided the questions into two sets, and each individual subject answered questions for one of the sets. The test itself was constructed by a DLPT testing expert and a senior native-speaking Arabic language instructor, using only the original Arabic documents and the Gold Standard translations. They had no access to any machine translation output during the test construction or scoring.

In August 2006, we administered the test at MIT to 49 test subjects who responded to announcements for paid experimental subjects. The subjects read the documents in a Latin square design, meaning that each subject saw each document, but only in one of the two conditions, randomly assigned. Subjects were allowed 5 hours to complete the test. Since the questions were divided into two sets for each document, the actual set of 49 subjects yielded approximately 25 "virtual subjects" reading the full list of 228 questions. The mean time spent on testing, not counting breaks or subject orientation, was 2.5 hours; fastest was 1.1 hours, slowest was 3.4 hours.

The subject responses were hand-graded by the two testing experts, following the pre-established answers in the test protocol. There was no pre-assessment of whether information was preserved or garbled in the MT when designing questions or responses in the test protocol. The testing experts were provided the reference translations and the original Arabic documents, but not the MT during scoring. Moreover, test conditions were masked in order to provide a blind assessment. The two testing experts provided both preliminary and final scores; multiple passes provided an opportunity to clarify the correct answers and to normalize scoring. The scoring agreement rate was 96% for the final scores.

4 Overall Results

The overall result for comprehension accuracy was 95% for subjects reading the Gold Standard translation and 74% for reading Machine Translation, across each of the genres and difficulty levels. The comprehension accuracy for each genre is shown in Figure 1. The two text genres score better than the audio genres, which is to be expected because the audio MT condition has more opportunities for error. Within each modality, the more standard, more structured genre fares better: newswire results are better than newsgroup results, and the more structured genre of broadcast news scores better than the less constrained, less structured conversations present in the talk radio shows.

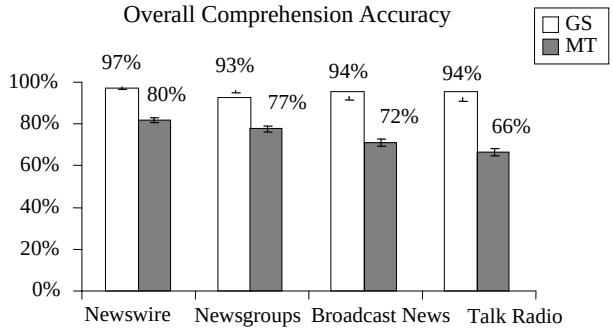


Figure 1. Comprehension Accuracy per Genre

The break-down by ILR level of difficulty for each question is shown in Figure 2. The general trend is consistent with what has been observed previously (Jones et al. 2005). The best results are at Level 2; Level 1 does well but not as well as expected. Thus the test has provided a key finding, which is that MT systems perform more poorly on Level 1, even when the data is matched to their capabilities. Level 3 is very challenging for the MT condition, and also more difficult in the GS condition. Using a standard 70 percent passing threshold, responses to questions on all MT documents, except for Level 3, received a passing grade.

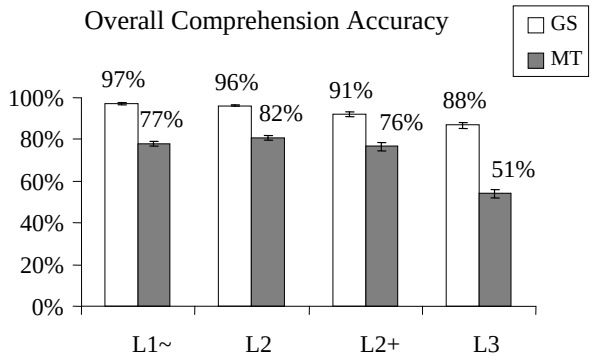


Figure 2. Comprehension Accuracy per Level.

To provide a snapshot of the ILR levels: L1 indicates sentence-level comprehensibility, and may include factual local announcements, etc.; L2 indicates paragraph-level comprehensibility; factual/ concrete, covering a wide spectrum of topics (politics, economy, society, culture, security, science); L3 involves extended discourse comprehensibility; the ability to understand hypotheses, supported opinion, implications, and abstract linguistic formulations, etc.

It was not possible to balance Level 3 documents across genres within the GALE evaluation data; except for those taken from Talk Radio, most documents did not reach that level of complexity. Hence, genre and difficulty level were not completely independent in this test.

5 Comprehension and Translation Error

We expect to see a relationship between comprehension rates and translation error. In an idealized case, we may expect a precise inverse correlation. We then compared comprehension rates with Human Translation Error Rate (HTER), an error measure for machine translation that counts the number of human edits required to change system MT output so that it contains all and only the information present in a Gold Standard reference (NIST, 2006). The linear regression line in Figure 3 shows the kind of inverse correlation we might expect. Subjects lose about 12% in comprehension for every 10% of translation error. The R^2 value is 33%. The low correlation suggests that the comprehension results are measuring a somewhat independent aspect of MT quality, which we feel is important. HTER does not directly address the facts that not all MT errors are equally important and that the texts contain inherent redundancy that the readers use to answer the questions. For exploratory purposes, we divide the graph of Figure 3 into four quadrants. Quadrant I and IV contain expected behavior: 122 data points of good translations and good comprehension results versus 43 points of bad translations and poor comprehension. Q-II has 24 robust points: the translations have high error, but somehow managed to contain enough well-translated words that people can answer the questions. Q-III has 28 fragile points: the few translation errors impaired comprehension.

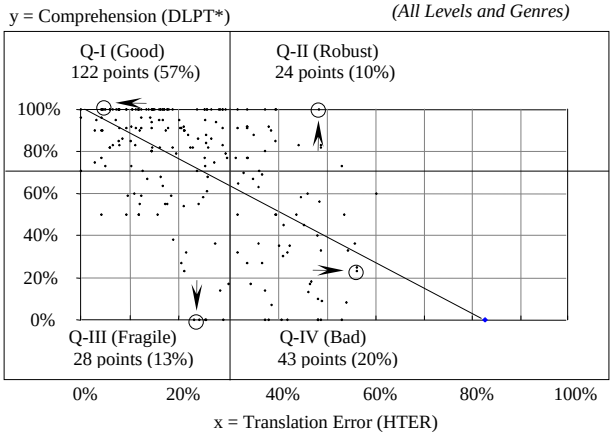


Figure 3. Comprehension vs. Translation Error.

We point out that there is a 1-to-1 mapping between comprehension questions and individual sub-passages of the documents in the data. Each point in Figure 3 plots the HTER of a single segment versus the average comprehension score on the corresponding question. The good and bad items are essentially a sanity-check on the experimental design. We expect to see good comprehension when translations are good, and we expect to see poor comprehension when translations are bad. Next we will examine the two other types: fragile and robust translations.

A fragile translation is one that has a good HTER score but a bad comprehension score. A sample fragile translation is one from a broadcast news which asks for a particular name: the HTER was a respectable 24%, but the MT comprehension accuracy was a flat 0%, since the name was missing. Everyone reading GS answered correctly.

A robust translation is one that has a bad HTER score but still manages to get a good comprehension score. A sample robust translation is one drawn from a posting providing instructions for foot massage. The text was quite garbled, with an HTER score of 48%, but the MT comprehension accuracy was a perfect 100%. Everyone reading the GS condition also answered the question correctly, which was that one should start a foot massage with oil. We note in passing that the highest error rate for a question with 100% comprehension is about 50%, shown with the up-arrow in Figure 3. We should be surprised to see any items with 100% comprehension for HTER rates above 50%, considering Shannon's estimate that written English is about 50% redundant. We expect that MT readers are making use of their general world knowledge to interpret the garbled MT output. A challenge is to identify robust translations, which are useful despite their high translation error rate.

6 Detailed Discussion

In this section we will discuss several aspects of the test in more detail: the scoring methodology, including a discussion of partial credit and inter-rater agreement; timing information; questions about personal names.

Each correct answer was assigned a score of 1, and each incorrect answer was assigned a score of 0. Partial credit was assigned on an ad-hoc basis, but normalized for scoring by assigning all non-integer scores to 0.5. This method yielded scores that were generally at the midpoint between binary scoring, in which non-integer scored were uniformly mapped either harshly to 0 or leniently to 1, the average difference between harsh and lenient scoring being approximately 11%. Inter-rater agreement was 96%.

The testing infrastructure we used recorded the amount of time spent on each document. The general trend is that people spend longer on MT than on GS. The mean percentage of time spent on MT compared with GS is 115% per item, meaning that it takes 15% more time to read MT than GS. The standard error was 4%. The median is 111%; minimum is 89% and maximum is 159%. In future analysis and experimentation we will conduct more fine-grained temporal estimates.

As we have seen in previous experiments, the performance for personal names is lower than for non-names. We observed that the name questions have 71% comprehension accuracy, compared with the 83% for questions about things other than personal names.

7 Conclusions and Future Work

We have long felt that Level 2 is the natural and successful level for machine translation. The ability to present concrete factual information that can be retrieved by the reader, without requirements for understanding the style, tone, or organizational pattern used by the writer seemed to be present in the previous work. It is worth pointing out that though we have many Level 1 questions, we are still not really testing Level 1 because the test does not contain true Level 1 documents. In future tests we wish to include Level 1 documents and questions.

Continuing along these lines, we are currently creating two new tests. We are constructing a new Arabic DLPT-star test, tailoring the document selection more specifically for comprehension testing and ensuring texts and tasks are at the intended ILR levels. We are also constructing a Mandarin Chinese test with similar design specifications. We intend for both of these tests to be available for a public machine translation evaluation to be conducted in 2007.

References

Doddington, G. 2002. *Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics*. Proceedings of HLT 2002.

NIST 2006. *GALE Go/No-Go Eval Plan*; www.nist.gov/speech/tests/gale/2006/doc/GALE06_evalplan.v2.pdf

Jones, D. A., W. Shen, et al. 2005a. *Measuring Translation Quality by Testing English Speakers with a New DLPT for Arabic.* Int'l Conf. on Intel. Analysis.

Interagency Language Roundtable Website. 2005. *ILR Skill Level Descriptions*: http://www.govtilr.org

Voss, Clare and Calandra Tate. 2006. *Task-based Evaluation of MT Engines*. European Association for Machine Translation conference.

White, JS and TA O'Connell. 1994. *Evaluation in the ARPA machine translation program: 1993 methodology*. Proceedings of the HLT workshop.

Semi-Supervised Learning for Semantic Parsing using Support Vector Machines

Rohit J. Kate and Raymond J. Mooney
Department of Computer Sciences
The University of Texas at Austin
1 University Station C0500
Austin, TX 78712-0233, USA
{rjkate,mooney}@cs.utexas.edu

Abstract

We present a method for utilizing unannotated sentences to improve a semantic parser which maps natural language (NL) sentences into their formal meaning representations (MRs). Given NL sentences annotated with their MRs, the initial supervised semantic parser learns the mapping by training Support Vector Machine (SVM) classifiers for every production in the MR grammar. Our new method applies the learned semantic parser to the unannotated sentences and collects unlabeled examples which are then used to retrain the classifiers using a variant of *transductive* SVMs. Experimental results show the improvements obtained over the purely supervised parser, particularly when the annotated training set is small.

1 Introduction

Semantic parsing is the task of mapping a natural language (NL) sentence into a complete, formal *meaning representation* (MR) which a computer program can execute to perform some task, like answering database queries or controlling a robot. These MRs are expressed in domain-specific unambiguous formal *meaning representation languages* (MRLs). Given a training corpus of NL sentences annotated with their correct MRs, the goal of a learning system for semantic parsing is to induce an efficient and accurate semantic parser that can map novel sentences into their correct MRs.

Several learning systems have been developed for semantic parsing, many of them recently (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Ge and Mooney, 2005; Kate and Mooney, 2006). These systems use supervised learning methods which only utilize annotated NL sentences. However, it requires considerable human effort to annotate sentences. In contrast, unannotated NL sentences are usually easily available. Semi-supervised learning methods utilize cheaply available unannotated data during training along with annotated data and often perform better than purely supervised learning methods trained on the same amount of annotated data (Chapelle et al., 2006). In this paper we present, to our knowledge, the first semi-supervised learning system for semantic parsing.

We modify KRISP, a supervised learning system for semantic parsing presented in (Kate and Mooney, 2006), to make a semi-supervised system we call SEMISUP-KRISP. Experiments on a real-world dataset show the improvements SEMISUP-KRISP obtains over KRISP by utilizing unannotated sentences.

2 Background

This section briefly provides background needed for describing our approach to semi-supervised semantic parsing.

2.1 KRISP: The Supervised Semantic Parsing Learning System

KRISP (Kernel-based Robust Interpretation for Semantic Parsing) (Kate and Mooney, 2006) is a supervised learning system for semantic parsing which

Proceedings of NAACL HLT 2007, Companion Volume, pages 81–84,
Rochester, NY, April 2007.

takes NL sentences paired with their MRs as training data. The productions of the formal MRL grammar are treated like semantic concepts. For each of these productions, a Support-Vector Machine (SVM) (Cristianini and Shawe-Taylor, 2000) classifier is trained using string similarity as the kernel (Lodhi et al., 2002). Each classifier can then estimate the probability of any NL substring representing the semantic concept for its production. During semantic parsing, the classifiers are called to estimate probabilities on different substrings of the sentence to compositionally build the most probable meaning representation (MR) of the sentence.

KRISP trains the classifiers used in semantic parsing iteratively. In each iteration, for every production π in the MRL grammar, KRISP collects positive and negative examples. In the first iteration, the set of positive examples for production π contains all sentences whose corresponding MRs use the production π in their parse trees. The set of negative examples includes all of the other training sentences. Using these positive and negative examples, an SVM classifier is trained for each production π using a string kernel. In subsequent iterations, the parser learned from the previous iteration is applied to the training examples and more refined positive and negative examples, which are more specific substrings within the sentences, are collected for training. Iterations are continued until the classifiers converge, analogous to iterations in EM (Dempster et al., 1977). Experimentally, KRISP compares favorably to other existing semantic parsing systems and is particularly robust to noisy training data (Kate and Mooney, 2006).

2.2 Transductive SVMs

SVMs (Cristianini and Shawe-Taylor, 2000) are state-of-the-art machine learning methods for classification. Given positive and negative training examples in some vector space, an SVM finds the maximum-margin hyperplane which separates them. Maximizing the margin prevents over-fitting in very high-dimensional data which is typical in natural language processing and thus leads to better generalization performance on test examples. When the unlabeled test examples are also available during training, a transductive framework for learning (Vapnik, 1998) can further improve the performance on the test examples.

Transductive SVMs were introduced in (Joachims, 1999). The key idea is to find the labeling of the test examples that results in the maximum-margin hyperplane that separates the positive and negative examples of *both* the training and the test data. This is achieved by including variables in the SVM's objective function representing labels of the test examples. Finding the exact solution to the resulting optimization problem is intractable, however Joachims (1999) gives an approximation algorithm for it. One drawback of his algorithm is that it requires the proportion of positive and negative examples in the test data be close to the proportion in the training data, which may not always hold, particularly when the training data is small. Chen et al. (2003) present another approximation algorithm which we use in our system because it does not require this assumption. More recently, new optimization methods have been used to scale-up transductive SVMs to large data sets (Collobert et al., 2006), however we did not face scaling problems in our current experiments.

Although transductive SVMs were originally designed to improve performance on the *test* data by utilizing its availability during training, they can also be directly used in a semi-supervised setting (Bennett and Demiriz, 1999) where unlabeled data is available during training that comes from the same distribution as the test data but is not the actual data on which the classifier is eventually to be tested. This framework is more realistic in the context of semantic parsing where sentences must be processed in real-time and it is not practical to re-train the parser transductively for every new test sentence. Instead of using an alternative semi-supervised SVM algorithm, we preferred to use a transductive SVM algorithm (Chen et al., 2003) in a semi-supervised manner, since it is easily implemented on top of an existing SVM system.

3 Semi-Supervised Semantic Parsing

We modified the existing supervised system KRISP, described in section 2.1, to incorporate semi-supervised learning. Supervised learning in KRISP involves training SVM classifiers on positive and negative examples that are substrings of the anno-

function TRAIN_SEMISUP_KRISP(Annotated corpus $\mathcal{A} = \{(s_i, m_i)|i = 1..N\}$, MRL grammar G,
Unannotated sentences $\mathcal{T} = \{t_i|i = 1..M\}$)
$\mathcal{C} \equiv \{C_\pi|\pi \in G\}$ = TRAIN_KRISP(A,G) // classifiers obtained by training KRISP
Let
$\mathcal{P} = \{p_\pi =$ Set of positive examples used in training $C_\pi|\pi \in G\}$
$\mathcal{N} = \{n_\pi =$ Set of negative examples used in training $C_\pi|\pi \in G\}$
$\mathcal{U} = \{u_\pi = \phi|\pi \in G\}$ // set of unlabeled examples for each production, initially all empty
for $i = 1$ **to** M **do**
$\{u^i_\pi|\pi \in G\}$ =COLLECT_CLASSIFIER_CALLS(PARSE($t_i, \mathcal{C}$))
$\mathcal{U} = \{u_\pi = u_\pi \cup u^i_\pi|\pi \in G\}$
for each $\pi \in G$ **do**
C_π =TRANSDUCTIVE_SVM_TRAIN(p_π, n_π, u_π) // retrain classifiers utilizing unlabeled examples
return classifiers $\mathcal{C} = \{C_\pi|\pi \in G\}$

Figure 1: SEMISUP-KRISP's training algorithm

tated sentences. In order to perform semi-supervised learning, these classifiers need to be given appropriate unlabeled examples. The key question is: Which substrings of the unannotated sentences should be given as unlabeled examples to which productions' classifiers? Giving all substrings of the unannotated sentences as unlabeled examples to all of the classifiers would lead to a huge number of unlabeled examples that would not conform to the underlying distribution of classes each classifier is trying to separate. SEMISUP-KRISP's training algorithm, described below and shown in Figure 1, addresses this issue.

The training algorithm first runs KRISP's existing training algorithm and obtains SVM classifiers for every production in the MRL grammar. Sets of positive and negative examples that were used for training the classifiers in the last iteration are collected for each production. Next, the learned parser is applied to the unannotated sentences. During the parsing of each sentence, whenever a classifier is called to estimate the probability of a substring representing the semantic concept for its production, that substring is saved as an unlabeled example for that classifier. These substrings are representative of the examples that the classifier will actually need to handle during testing. Note that the MRs obtained from parsing the unannotated sentences do not play a role during training since it is unknown whether or not they are correct. These sets of unlabeled examples for each production, along with the sets of positive and negative examples collected earlier, are then used to retrain the classifiers using transductive SVMs. The retrained classifiers are finally returned and used in the final semantic parser.

4 Experiments

We compared the performance of SEMISUP-KRISP and KRISP in the GEOQUERY domain for semantic parsing in which the MRL is a functional language used to query a U.S. geography database (Kate et al., 2005). This domain has been used in most of the previous work. The original corpus contains 250 NL queries collected from undergraduate students and annotated with their correct MRs (Zelle and Mooney, 1996). Later, 630 additional NL queries were collected from real users of a web-based interface and annotated (Tang and Mooney, 2001). We used this data as *unannotated* sentences in our current experiments. We also collected an additional 407 queries from the same interface, making a total of 1,037 unannotated sentences.

The systems were evaluated using standard 10-fold cross validation. All the unannotated sentences were used for training in each fold. Performance was measured in terms of precision (the percentage of generated MRs that were correct) and recall (the percentage of all sentences for which correct MRs were obtained). An output MR is considered correct if and only if the resulting query retrieves the same answer as the correct MR when submitted to the database. Since the systems assign confidences to the MRs they generate, the entire range of the precision-recall trade-off can be obtained for a system by measuring precision and recall at various confidence levels. We present learning curves for the best F-measure (harmonic mean of precision and re-

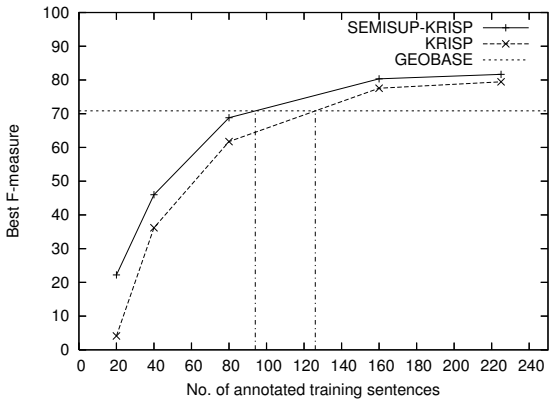


Figure 2: Learning curves for the best F-measures on the GEOQUERY corpus.

call) obtained across the precision-recall trade-off as the amount of annotated training data is increased. Figure 2 shows the results for both systems.

The results clearly show the improvement SEMISUP-KRISP obtains over KRISP by utilizing unannotated sentences, particularly when the number of annotated sentences is small. We also show the performance of a hand-built semantic parser GEOBASE (Borland International, 1988) for comparison. From the figure, it can be seen that, on average, KRISP achieves the same performance as GEOBASE when it is given 126 annotated examples, while SEMISUP-KRISP reaches this level given only 94 annotated examples, a 25.4% savings in human-annotation effort.

5 Conclusions

This paper has presented a semi-supervised approach to semantic parsing. Our method utilizes unannotated sentences during training by extracting unlabeled examples for the SVM classifiers it uses to perform semantic parsing. These classifiers are then retrained using transductive SVMs. Experimental results demonstrated that this exploitation of unlabeled data significantly improved the accuracy of the resulting parsers when only limited supervised data was provided.

Acknowledgments

This research was supported by a Google research grant. The experiments were run on the Mastodon cluster provided by NSF grant EIA-0303609.

References

K. Bennett and A. Demiriz. 1999. Semi-supervised support vector machines. *Advances in Neural Information Processing Systems*, 11:368–374.

Borland International. 1988. *Turbo Prolog 2.0 Reference Guide*. Borland International, Scotts Valley, CA.

O. Chapelle, B. Schölkopf, and A. Zien, editors. 2006. *Semi-Supervised Learning*. MIT Press, Cambridge, MA.

Y. Chen, G. Wang, and S. Dong. 2003. Learning with progressive transductive support vector machine. *Pattern Recognition Letters*, 24:1845–1855.

R. Collobert, F. Sinz, J. Weston, and L. Bottou. 2006. Large scale transductive SVMs. *Journal of Machine Learning Research*, 7(Aug):1687–1712.

N. Cristianini and J. Shawe-Taylor. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38.

R. Ge and R. J. Mooney. 2005. A statistical semantic parser that integrates syntax and semantics. In *Proc. of CoNLL-05*, pages 9–16, Ann Arbor, MI, July.

T. Joachims. 1999. Transductive inference for text classification using support vector machines. In *Proc. of ICML-99*, pages 200–209, Bled, Slovenia, June.

R. J. Kate and R. J. Mooney. 2006. Using string-kernels for learning semantic parsers. In *Proc. of COLING/ACL-06*, pages 913–920, Sydney, Australia, July.

R. J. Kate, Y. W. Wong, and R. J. Mooney. 2005. Learning to transform natural to formal languages. In *Proc. of AAAI-05*, pages 1062–1068, Pittsburgh, PA, July.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444.

L. R. Tang and R. J. Mooney. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Proc. of ECML-01*, pages 466–477, Freiburg, Germany.

V. N. Vapnik. 1998. *Statistical Learning Theory*. John Wiley & Sons.

J. M. Zelle and R. J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proc. of AAAI-96*, pages 1050–1055, Portland, OR, August.

L. S. Zettlemoyer and M. Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proc. of UAI-05*, Edinburgh, Scotland, July.

Discriminative Alignment Training without Annotated Data for Machine Translation

Patrik Lambert, Rafael E. Banchs and Josep M. Crego
TALP Research Center
Jordi Girona Salgado 1–3
08034 Barcelona, Spain
{lambert, rbanchs, jmcrego}@gps.tsc.upc.edu

Abstract

In present Statistical Machine Translation (SMT) systems, alignment is trained in a previous stage as the translation model. Consequently, alignment model parameters are not tuned in function of the translation task, but only indirectly. In this paper, we propose a novel framework for discriminative training of alignment models with automated translation metrics as maximization criterion. In this approach, alignments are optimized for the translation task. In addition, no link labels at the word level are needed. This framework is evaluated in terms of automatic translation evaluation metrics, and an improvement of translation quality is observed.

1 Introduction

In the first SMT systems (Brown et al., 1993), word alignment was introduced as a hidden variable of the translation model. When word-based translation models have been replaced by phrase-based models (Zens et al., 2002), alignment[1] and translation model training have become two separated tasks.

The system of Brown *et al.* was based on the noisy channel approach. Present SMT systems use a more general maximum entropy approach in which a log-linear combination of multiple feature functions is implemented (Och and Ney, 2002). Within this new framework translation quality can be tuned by adjusting the weight of each feature function in the log-linear combination. In order to improve translation quality, this tuning can be effectively performed by minimizing translation error over a development corpus for which manually translated references are available (Och, 2003). As a separate first stage of the process, alignment is not in practice directly tuned in function of the machine translation task.

Tuning alignment for an MT system is subject to practical difficulties. Unsupervised systems (Och and Ney, 2003; Liang et al., 2006) are based on generative models trained with the EM algorithm. They require large computational resources, and incorporating new features is difficult. In contrast, adding new features to some supervised systems (Liu et al., 2005; Moore, 2005; Ittycheriah and Roukos, 2005) is easy, but the need of annotated data is a problem.

A more general difficulty, however, is that of finding an alignment evaluation metric favoring alignments which benefit Machine Translation. The fact that the required alignment characteristics depend on each particular system makes it even more difficult. It seems that high precision alignments are better for phrase-based SMT (Chen and Federico, 2006; Ayan and Dorr, 2006), whereas high recall alignments are more suited to N-gram SMT (Mariño et al., 2006). In this context, alignment quality improvements does not necessarily imply translation quality improvements. This is in agreement with the observation of a poor correlation between word alignment error rate (AER (Och and Ney, 2000)) and automatic translation evaluation metrics (Ittycheriah and Roukos, 2005; Vilar et al., 2006).

[1] Hereinafter, alignment will refer to word alignment, unless otherwise stated.

Proceedings of NAACL HLT 2007, Companion Volume, pages 85–88,
Rochester, NY, April 2007.

Recently some alignment evaluation metrics have been proposed which are more informative when the alignments are used to extract translation units (Fraser and Marcu, 2006; Ayan and Dorr, 2006). However, these metrics assess translation quality very indirectly.

In this paper, we propose a novel framework for discriminative training of alignment models with automated translation metrics as maximization criterion. Thus we just need a reference aligned at the sentence level instead of link labels at the word level.

The paper is structured as follows. Section 2 explains the models used in our word aligner, focusing on the features designed to account for the specificities of the SMT system. In section 3, our minimum error training procedure is described and experimental results are shown. Finally, some concluding remarks and lines of further research are given.

2 Bilingual Word Aligner

For versatility and efficiency requirements, we implemented BIA, a BIlingual word Aligner similar to that of Moore (2005). BIA consists in a beam-search decoder searching, for each sentence pair, the alignment which minimizes the cost of a linear combination of various models. The differences with the system of Moore lie in the features, which we specially designed to suit our translation system (N-gram SMT (Mariño et al., 2006)). Its particularity is the translation model, which is based on a 4-gram language model of bilingual units referred to as tuples. Two issues regarding this translation model can be dealt with at the alignment stage.

Firstly, in order to estimate the bilingual n-gram model, only one monotonic segmentation of each sentence pair is performed. Thus long reorderings cause long and sparse tuples to be extracted. For example, if the first source word is linked to the last target word, only one tuple can be extracted, which contains the whole sentence pair. This kind of tuple is not reusable, and the data between its two extreme words are lost.

Secondly, it occurs very often that unlinked words (*i.e.* linked to NULL) end up producing tuples with NULL source sides. This cannot be allowed since no NULL is expected to occur in a translation input. This problem is solved by preprocessing alignments before tuple extraction such that any unlinked target word is attached to either its precedent or its following word.

Taking theses issues into account, we implemented the following features:

- distinct source and target unlinked word penalties: since unlinked words have a different impact whether they appear in the source or target language, we introduced an unlinked word feature for each side of the sentence pair.
- link bonus: in order to accommodate the N-gram model preference for higher recall alignment, we introduced a feature which adds a bonus for each link in the alignment.
- embedded word position penalty: this feature penalizes situations like the one depicted in figure 1. In this example, the bilingual units s2-t2 and s3-t3 cannot be extracted because word positions s2 and s3 are embedded between links s1-t1 and s4-t1. Thus the link s4-t1 may introduces data sparseness in the translation model, although it may be a correct link. So we want to have a feature which counts the number of embedded word positions in an alignment.

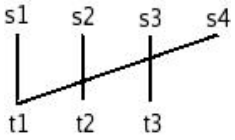


Figure 1: Word positions embedded in a tuple.

In addition to the embedded word position feature, we used the same two distortion features as Moore to penalize reorderings in the alignment (one sums the number of crossing links, and the other one sums the amplitude of crossing links). We also used the ϕ^2 score (Gale and Church, 1991) as a word association model, and as a POS-tags association model.

3 Experimental Work

For these experiments we used the Chinese-English data provided for IWSLT'06 evaluation campaign (Paul, 2006). The training set contains 46000 sentences (of 6.7 and 7.0 average length). Parameters were tuned over the development set (dev4) provided, consisting of 489 sentences of 11.2 words in average, with 7 references. Our test set was a selection of 500 sentences (of 6 words in average, with 16 references) among dev1, dev2 and dev3 sets.

3.1 Optimization Procedure

Once the alignment models were computed, a set of optimal log-linear coefficients was estimated via the optimization procedure depicted in Figure 2.

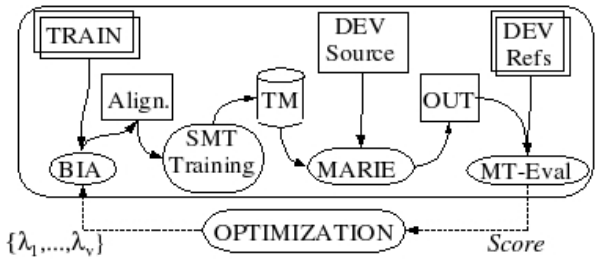


Figure 2: Optimization loop.

The training corpus was aligned with a set of initial parameters $\lambda_1, \ldots, \lambda_7$. This alignment was used to extract tuples and build a bilingual N-gram translation model (TM). A baseline SMT system, consisting of MARIE decoder and this translation model as unique feature[2], was used to produce a translation (OUT) of the development source set. Then, translation quality over the development set is maximized by iteratively varying the set of coefficients.

The optimization procedure was performed by using the SPSA algorithm (Spall, 1992). SPSA is a stochastic implementation of the conjugate gradient method which requires only two evaluations of the objective function. It was observed to be more robust than the Downhill Simplex method when tuning SMT coefficients (Lambert and Banchs, 2006).

Each function evaluation required to align the training corpus and build a new translation model. The algorithm converged after about 80 evaluations, lasting each 17 minutes with a 3 GHz processor. Alignment decoding was performed with a beam of 10 (it took 50 seconds and required 8 MB memory).

Finally, the corpus was aligned with the optimum set of coefficients, and a full SMT system was build, with a target language model (trained on the provided training data), a word bonus model and two lexical models. SMT models weights were optimized with a standard Minimum Error Training (MET) strategy[3] and the test corpus was translated with the full system. To contrast the results, full translation systems were also build extracting tuples from various combinations of GIZA++ alignments (trained with 50 classes and respectively 4,5 and 4 iterations of models 1,HMM and 4). In order to limit the error introduced by MET, we translated the test corpus with three sets of SMT model weights, and took the average and standard deviation.

3.2 Results

Table 1 shows results obtained with the full SMT system on the test corpus, with GIZA++ alignments, and BIA alignments optimized in function of three metrics: BLEU, NIST, and BLEU+4*NIST. The standard deviation is indicated in parentheses. Although results for systems trained with different BIA alignments present more variability than systems trained with GIZA++ alignments, they achieve better average scores, and one of them obtains much higher scores. Unexpectedly, BIA alignments tuned with NIST yield the system with worse NIST score.

4 Conclusions and further work

We proposed a novel framework for discriminative training of alignment models with automated translation metrics as maximization criterion. According to this type of metrics, the translation systems trained from the optimized alignments clearly performed better than the ones trained from Giza++ alignment combinations.

In addition, this first version of the alignment system has very basic models and could be improved. We could certainly improve the association score model, for example adding discount factors or adding more association score types, or dictionaries.

During the alignment coefficient optimization depicted in Figure 2, only the baseline SMT system is used. In future work, we could consider using various SMT features (as would be required for a phrase-based SMT system).

Our approach, as it is, cannot be applied to a large corpus, since it requires to align the whole training corpus at each iteration. Thus an interesting further research would consist in determining whether the

[2] An N-gram SMT system can produce good translations without additional target language model since the target language is modeled inside the bilingual N-gram model.

[3] SMT parameters are not optimized together with alignment parameters for two main reasons. Firstly, translation is more sensitive to variations of SMT parameters. Secondly, alignment is optimized over the full training set, whereas SMT is tuned over the development set.

System	BLEU	NIST	PER	WER
GIZA++ union	42.7 (1.1)	8.82 (0.07)	34.7 (0.2)	43.7 (0.4)
GIZA++ intersection	42.4 (0.9)	8.53 (0.07)	37.0 (0.9)	45.0 (1.3)
GIZA++ Zh→En	43.7 (0.9)	8.90 (0.2)	37.2 (1.4)	45.5 (2.0)
BIA (BLEU)	44.8 (0.4)	9.00 (0.04)	35.7 (0.07)	43.8 (0.09)
BIA (BLEU+4*NIST)	47.0 (1.5)	8.83 (0.4)	32.9 (0.8)	40.9 (0.5)
BIA (NIST)	44.8 (0.1)	8.55 (0.14)	33.0 (0.2)	41.4 (0.5)

Table 1: *Automatic translation evaluation results.*

alignment parameters trained on a part of the corpus are valid for the whole corpus.

Finally, some Giza++ parameters may also be tuned, in the same way as for BIA parameters.

5 Acknowledgments

This work has been partially funded by the European Union under the integrated project TC-STAR - Technology and Corpora for Speech to Speech Translation -(IST-2002-FP6-506738, http://www.tc-star.org) and by the Spanish Government under grant TEC2006-13964-C03 (AVIVAVOZ project).

References

Necip F. Ayan and Bonnie J. Dorr. 2006. Going Beyond AER: An Extensive Analysis of Word Alignments and Their Impact on MT. In *Proc. COLING-ACL*, pages 9–16, Sydney, Australia.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Boxing Chen and Marcello Federico. 2006. Improving phrase-based statistical translation through combination of word alignment. In *Proc. FinTAL*, Turku, Finland.

Alexander Fraser and Daniel Marcu. 2006. Semi-supervised training for statistical word alignment. In *Proc. COLING-ACL*, pages 769–776, Sydney, Australia.

W. Gale and K. W. Church. 1991. Identifying word correspondences in parallel texts. In *DARPA Speech and Natural Language Workshop*, Asilomar, CA.

Abraham Ittycheriah and Salim Roukos. 2005. A maximum entropy word aligner for arabic-english machine translation. In *Proc. HLT-EMNLP*, pages 89–96, Vancouver, Canada.

Patrik Lambert and Rafael E. Banchs. 2006. Tuning Machine Translation Parameters with SPSA. In *Proc. IWSLT*, pages 190–196, Kyoto, Japan.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proc. the HLT-NAACL*, pages 104–111, New York City, USA.

Yang Liu, Qun Liu, and Shouxun Lin. 2005. Log-linear models for word alignment. In *Proc. ACL*, pages 459–466, Ann Arbor, Michigan.

José B. Mariño, Rafael E. Banchs, Josep M. Crego, Adrià de Gispert, Patrik Lambert, José A.R. Fonollosa, and Marta R. Costa-jussà. 2006. N-gram based machine translation. *Computational Linguistics*, 32(4):527–549.

Robert C. Moore. 2005. A discriminative framework for bilingual word alignment. In *Proc. HLT-EMNLP*, pages 81–88, Vancouver, Canada.

Franz Josef Och and Hermann Ney. 2000. A comparison of alignment models for statistical machine translation. In *Proc. COLING*, pages 1086–1090, Saarbrucken,Germany.

F.J. Och and H. Ney. 2002. Dicriminative training and maximum entropy models for statistical machine translation. In *Proc. ACL*, pages 295–302, Philadelphia, PA.

F.J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, March.

F.J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. ACL*, pages 160–167.

Michael Paul. 2006. Overview of the IWSLT 2006 Evaluation Campaign. In *Proc. IWSLT*, pages 1–15, Kyoto, Japan.

James C. Spall. 1992. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Trans. Automat. Control*, 37:332–341.

David Vilar, Maja Popovic, and Hermann Ney. 2006. AER: Do we need to "improve" our alignments? In *Proc. IWSLT*, pages 205–212, Kyoto, Japan.

R. Zens, F.J. Och, and H. Ney. 2002. Phrase-based statistical machine translation. In Springer Verlag, editor, *Proc. German Conf. on Artificial Intelligence (KI)*.

A Geometric Interpretation of Non-Target-Normalized Maximum Cross-channel Correlation for Vocal Activity Detection in Meetings

Kornel Laskowski
interACT, Universität Karlsruhe
Karlsruhe, Germany
kornel@ira.uka.de

Tanja Schultz
interACT, Carnegie Mellon University
Pittsburgh PA, USA
tanja@cs.cmu.edu

Abstract

Vocal activity detection is an important technology for both automatic speech recognition and automatic speech understanding. In meetings, standard vocal activity detection algorithms have been shown to be ineffective, because participants typically vocalize for only a fraction of the recorded time and because, while they are not vocalizing, their channels are frequently dominated by crosstalk from other participants. In the present work, we review a particular type of normalization of maximum cross-channel correlation, a feature recently introduced to address the crosstalk problem. We derive a plausible geometric interpretation and show how the frame size affects performance.

1 Introduction

Vocal activity detection (VAD) is an important technology for any application with an automatic speech recognition (ASR) front end. In meetings, participants typically vocalize for only a fraction of the recorded time. Their temporally contiguous contributions should be identified prior to ASR in order to leverage speaker adaptation schemes and language model constraints, and to associate recognized output with specific speakers (who said what). Segmentation into such contributions is informed primarily by VAD on a frame-by-frame basis.

Individual head-mounted microphone (IHM) recordings of meetings present a particular challenge for VAD, due to crosstalk from other participants. Most state-of-the-art VAD systems for meetings rely on decoding in a binary speech/non-speech space, assuming independence among participants, but are increasingly relying on features specifically designed to address the crosstalk issue (Wrigley et al., 2005).

A feature which has attracted attention since its use in VAD post-processing in (Pfau et al., 2001) is the maximum cross-channel correlation (XC), $\max_\tau \phi_{jk}(\tau)$, between channels j and k, where τ is the lag. When designing features descriptive of the kth channel, XC is frequently normalized by the energy in the target[1] channel k (Wrigley et al., 2003). Alternately, XC can be normalized by the energy in the non-target channel j (Laskowski et al., 2004), a normalization which we refer to here as NT-Norm, extending the Norm and S-Norm naming conventions in (Wrigley et al., 2005). Table 1 shows several types of normalizations which have been explored.

Normalization of XC		Mean	Min	Max
(none)	$\max_{j\neq k} \phi_{jk}(\tau)$	**[2][4]**	[2][4]	[2][4]
Norm	$\frac{\max_{j\neq k} \phi_{jk}(\tau)}{\phi_{kk}(0)}$	**[2][4]**	[2][4]	**[2][4]**
S-Norm	$\frac{\max_{j\neq k} \phi_{jk}(\tau)}{\sqrt{\phi_{jj}(0)\phi_{kk}(0)}}$	[2]**[4]**[5]	[2][4]	[1][2]**[4]**
NT-Norm	$\frac{\max_{j\neq k} \phi_{jk}(\tau)}{\phi_{jj}(0)}$	[3]	[6]	[6]

Table 1: Normalizations and statistics of cross-channel correlation features to describe channel k. In [1], a median-smoothed version was used in post-processing. In [3], the sum (JMXC) was used instead of the mean. In [5], cross-correlation was computed over samples and features. In [6], the minimum and the maximum were jointly referred to as NMXC. References in bold depict features selected by an automatic feature selection algorithm in [2] and [4]. (1:(Pfau et al., 2001), 2:(Wrigley et al., 2003), 3:(Laskowski et al., 2004), 4:(Wrigley et al., 2005), 5:(Huang, 2005), 6:(Boakye and Stolcke, 2006))

[1]The target/non-target terms are due to (Boakye and Stolcke, 2006).

Proceedings of NAACL HLT 2007, Companion Volume, pages 89–92,
Rochester, NY, April 2007.

The present work revisits NT-Norm normalization, which has been successfully used in a threshold detector (Laskowski et al., 2004), in automatic initial label assignment (Laskowski and Schultz, 2006), and as part of a two-state decoder feature vector (Boakye and Stolcke, 2006). Our main contribution is a geometric interpretation of NT-Norm XC, in Section 2. We also describe, in Section 3, several contrastive experiments, and discuss the results in Section 4.

2 Geometric Interpretation

We propose an interpretable geometric approximation to NT-Norm XC for channel k,

$$\xi_{k,j} \;=\; \frac{\max_\tau \phi_{jk}(\tau)}{\phi_{jj}}\,, \quad \forall j{\neq}k \tag{1}$$

We assume the simplified response in the kth IHM microphone at a distance d_k from a single point source $s(t)$ to be

$$m_k(t) \;\doteq\; A_k\left(\frac{1}{d_k}\,s\left(t-\frac{d_k}{c}\right)+\eta_k(t)\right), \tag{2}$$

where c, A_k and $\eta_k(t)$ are the speed of sound, the gain of microphone k, and source-uncorrelated noise at microphone k, respectively. Cross-channel correlation is approximated over a frame of size Ω by

$$\phi_{jk}(\tau) \;=\; \int_\Omega \frac{A_j A_k}{d_j d_k} s(t)\, s(t-\tau)\, dt\,, \tag{3}$$

where $\tau \equiv (d_j - d_k)/c$. Letting $\mathcal{P}_s \equiv \int_\Omega s^2(t)\,dt$ and $\mathcal{P}_{\eta_k} \equiv \int_\Omega \eta_k^2(t)\,dt$,

$$\phi_{jj}(0) \;=\; A_j^2\left(\frac{1}{d_j^2}\,\mathcal{P}_s + \mathcal{P}_{\eta_j}\right)\,, \tag{4}$$

$$\max_\tau \phi_{jk}(\tau) \;=\; \frac{A_j A_k}{d_j d_k}\mathcal{P}_s\,, \tag{5}$$

respectively, as the maximum of $\phi_{jk}(\tau)$ occurs at $\tau^* = (d_k - d_j)/c$. In consequence,

$$\frac{\max_\tau \phi_{jk}(\tau)}{\phi_{jj}(0)} \;\approx\; \frac{d_j}{d_k}\,, \tag{6}$$

provided that

$$\frac{A_k}{A_j}\left[1-\frac{\mathcal{P}_{\eta_j}}{\frac{1}{d_j^2}\mathcal{P}_s+\mathcal{P}_{\eta_j}}\right] \;\approx\; 1\,, \tag{7}$$

i.e., under assumptions of similar microphone gains, a non-negligible farfield signal-to-noise ratio at each microphone, and the simplifications embodied in Equation 2, NT-Norm XC approximates the relative distances of 2 microphones to the single point source $s(t)$. We stress that this approximation requires no side knowledge about the true positions of the participants or of their microphones.

Importantly, this interpretation is valid only if τ^* lies within the integration window Ω in Equation 3. In (Boakye and Stolcke, 2006), the authors showed that when the analysis window is 25 ms, the NMXC feature is not as robust as frame-level energy flooring followed by cross-channel normalization (NLED).

3 Experimental Setup

3.1 VAD and ASR Systems

Our multispeaker VAD system, shown in Figure 1, was introduced in (Laskowski and Schultz, 2006). Rather than detecting the 2-state speech ($\mathcal{V}$) vs. non-speech ($\mathcal{N}$) activity of each partipant independently, the system implements a Viterbi search for the best path through a 2^K-state vocal interaction space, where K is the number of participants. Segmentation consists of three passes: initial label assignment (ILA), described in the next subsection, for acoustic model training; simultaneous multi-participant Viterbi decoding; and smoothing to produce segments for ASR. In the current work, during decoding, we limit the maximum number of simultaneously vocalizing participants to 3.

This system is an improved version of that fielded in the NIST Rich Transcription 2006 Meeting Recognition evaluation (RT06s)[2], to produce automatic segmentation in the IHM condition on conference meetings. The ASR system which we use in this paper is as described in (Fügen et al., 2007).

3.2 Unsupervised ILA

For unsupervised labeling of the test audio, prior to acoustic model training, we employ the criterion

$$\tilde{\mathbf{q}}[k] \;=\; \begin{cases} \mathcal{V} & \text{if } \sum_{j\neq k} \log\left(\frac{\max_\tau \phi_{jk}(\tau)}{\phi_{jj}(0)}\right) > 0 \\ \mathcal{N} & \text{otherwise} \quad . \end{cases} \tag{8}$$

Assuming equality in Equation 6, this corresponds to *declaring a participant as vocalizing when the distance between the location of the dominant sound source and that participant's microphone is smaller than the geometric mean of the distances from the source to the remaining microphones*, ie. when

$$\sqrt[K-1]{\prod_{j\neq k} d_j} \;>\; d_k \tag{9}$$

[2] `http://www.nist.gov/speech/tests/rt/`

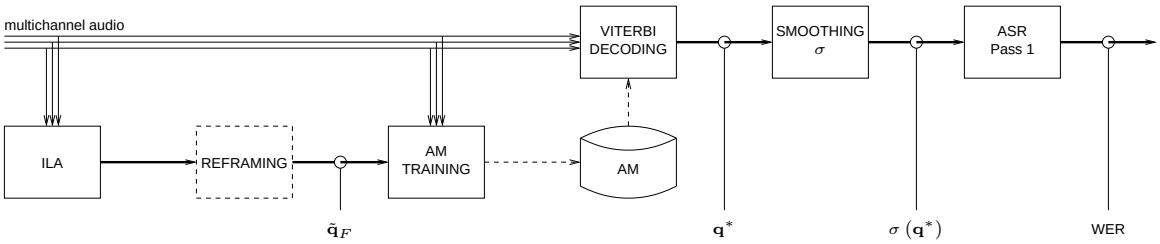


Figure 1: VAD system architecture, with 4 error measurement points. Symbols as in the text.

We refer to this algorithm as ILAave. For contrast we also consider ILAmin, with the sum in Equation 8 replaced by the minimum over $j \neq k$. This corresponds to *declaring a participant as vocalizing when the distance between the location of the dominant sound source and that participant's microphone is smaller than the distance from the source to any other microphone*. We do not consider ILAmax, whose interpretation in light of Equation 6 is not useful.

3.3 Data

The data used in the described experiments consist of two datasets from the NIST RT-05s and RT-06s evaluations. The data which had been used for VAD system improvement, `rt05s_eval*`, is the complete `rt05s_eval` set less one meeting, `NIST_20050412-1303`. This meeting was excluded as it contains a participant without a microphone, a condition known a priori to be absent in `rt06s_eval`; we use the latter in its entirety.

3.4 Description of Experiments

The experiments we present aim to compare ILAave and ILAmin, and to show how the size of the integration window, Ω, affects system performance. As our VAD decoder operates at a frame size of 100ms, we introduce a reframing step between the ILA component and both AM training and decoding; see Figure 1. $\mathcal{V}$ is assigned to each 100ms frame if 50% or more of the frame duration is assigned $\mathcal{V}$ by ILA; otherwise, the 100ms frame is assigned an $\mathcal{N}$ label.

We measure performance in four locations within the combined VAD+ASR system architecture, also shown in Figure 1. We compute a VAD frame error just after reframing ($\tilde{\mathbf{q}}_F$), just after decoding ($\mathbf{q}^*$), and just after smoothing ($\sigma(\mathbf{q}^*)$). This error is the sum of the miss rate (MS), and the false alarm rate excluding intervals of all-participant silence (FAX), computed against unsmoothed word-level forced alignment references. We use this metric for comparative purposes only, across the various measurement points. We also use first-pass ASR word error rates (WERs), after lattice rescoring, as a final measure of performance impact.

We evaluate, over a range of ILA frame sizes, the performance of ILAave(3), with a maximum number of simultaneously vocalizing participants of 3, and for the contrastive ILAmin. We note that ILAmin is capable of declaring at most one microphone at a time as being worn by a current speaker. As a result, construction of acoustic models for overlapped vocal activity states, described in (Laskowski and Schultz, 2006), results in states of at most 2 simultaneously vocalizing participants. We therefore refer to ILAmin as ILAmin(2), and additionally consider ILAave(2), in which states with 3 simultaneously vocalizing participants are removed.

4 Results and Discussion

We show the results of our experiments in Table 2. First-pass WERs, using reference segmentation (`.stm`), vary by 1.3% absolute (abs) between `rt05s_eval` and `rt06s_eval`. We also note that removing the one meeting with a participant without a microphone reduces the `rt05s_eval` manual segmentation WER by 1.7% abs. WERs obtained with automatic segmentation should be compared to the manual segmentation WERs for each set.

As the $\tilde{\mathbf{q}}_F$ columns shows, ILAmin(2) entails significantly more VAD errors than ILAave. Notably, although we do not show the breakdown, ILAmin(2) is characterized by fewer false alarms, but misses much more speech than ILAave(2). This is due in part to its inability to identify simultaneous talkers. However, following acoustic model training and use ($\mathbf{q}^*$), the VAD error rates between the two algorithms are approximately equal.

In studying the WERs for each ILA algorithm independently, the variation across ILA frame sizes in the range 25–100 ms can be significant: for example, it is 1.2% abs for ILAmin(2) on `rt06s_eval`, compared to the difference with manual segmentation of 3.1% abs. Error curves, as a function of ILA frame size, are predominantly shallow parabolas, except at 75 ms (notably for ILAmin(2) at $\tilde{\mathbf{q}}_F$); we believe that

ILA	Ω	VAD, rt05s			WER, 1st pass		
		$\tilde{\mathbf{q}}_F$	$\mathbf{q}^*$	$\sigma(\mathbf{q}^*)$	05	05*	06
ave3	100	**31.3**	16.7	16.0	39.0	34.1	39.6
	75	33.6	**16.6**	**15.9**	38.9	34.1	39.9
	50	35.2	16.7	16.0	**38.8**	**34.0**	**39.3**
	25	36.8	17.3	16.3	39.6	34.2	39.7
ave2	100	**31.3**	15.8	15.2	37.8	34.4	39.7
	75	33.6	**15.6**	**15.0**	37.9	34.4	39.6
	50	35.2	15.8	15.2	**37.6**	**34.3**	**39.3**
	25	36.8	16.4	15.6	38.1	**34.3**	39.5
min2	100	**43.4**	15.8	14.7	38.2	35.2	**39.3**
	75	51.9	**15.6**	**14.6**	38.1	35.2	**39.3**
	50	47.1	15.7	**14.6**	**37.9**	**35.1**	40.1
	25	47.7	16.2	14.9	38.1	35.4	40.5
refs		9.5	9.5	9.5	36.1	34.4	37.4

Table 2: VAD errors, measured at three points in our system, and first-pass WERs for `rt05s_eval` (`05`), as well as first-pass WERs for `rt05s_eval*` (`05*`) and `rt06s_eval` (`06`). Results are shown for 3 contrastive VAD systems (ILAave(3), ILAave(2) and ILAmin(2)), and 4 ILA frame sizes (100ms, 75ms, 50ms, and 25ms).

this is because 75 ms does not divide evenly into the decoder frame size of 100 ms, causing more deletions across the reframing step than for other ILA frame sizes. Error minima appear for an ILA frame size somewhere between 50 ms and 75 ms, for both ASR and post-decoding VAD errors.

Although (Pfau et al., 2001) considered a maximum lag of 250 samples (15.6ms, or 5m at the speed of sound), their computation of S-Norm XC used a rectangular window. Here, as in (Laskowski and Schultz, 2006) and (Boakye and Stolcke, 2006), we use a Hamming window. Our results suggest that a large, broadly tapered window is important for Equation 6 to hold.

The table also shows that for datasets without uninstrumented participants, `rt05s_eval*` and `rt06s_eval`, ILAmin(2) is outperformed by ILAave(2) by as much as 1.1% abs in WER, especially at small frame sizes. The difference for the full `rt05s_eval` dataset is smaller. The results also suggest that reducing the maximum degree of simultaneous vocalization from 3 to 2 during decoding is an effective means of reducing errors (ASR insertions, not shown) for uninstrumented participants.

5 Conclusions

We have derived a geometric approximation for a particular type of normalization of maximum cross-channel correlation, NT-Norm XC, recently introduced for multispeaker vocal activity detection. Our derivation suggests that it is effectively comparing the distance between each speaker's mouth and each microphone. This is novel, as geometry is most often inferred using the lag of the crosscorrelation maximum, rather than its amplitude.

Our experiments suggest that frame sizes of 50–75 ms lead to WERs which are lower than those for either 100 ms or 25 ms by as much as 1.2% abs; that ILAave outperforms ILAmin as an initial label assignment criterion; and that reducing the degree of simultaneous vocalization during decoding may address problems due to uninstrumented participants.

6 Acknowledgments

This work was partly supported by the European Union under the integrated project CHIL (IST-506909), Computers in the Human Interaction Loop.

References

K. Boakye and A. Stolcke. 2006. Improved Speech Activity Detection Using Cross-Channel Features for Recognition of Multiparty Meetings. *Proc. of INTERSPEECH*, Pittsburgh PA, USA, pp1962–1965.

C. Fügen, S. Ikbal, F. Kraft, K. Kumatani, K. Laskowski, J. McDonough, M. Ostendorf, S. Stüker, and M. Wölfel. 2007. The ISL RT-06S Speech-to-Text Evaluation System. *Proc. of MLMI*, Springer Lecture Notes in Computer Science **4299**, pp407–418.

Z. Huang and M. Harper. 2005. Speech Activity Detection on Multichannels of Meeting Recordings. *Proc. of MLMI*, Springer Lecture Notes in Computer Science **3869**, pp415–427.

K. Laskowski, Q. Jin, and T. Schultz. 2004. Crosscorrelation-based Multispeaker Speech Activity Detection. *Proc. of INTERSPEECH*, Jeju Island, South Korea, pp973–976.

K. Laskowski and T. Schultz. 2006. Unsupervised Learning of Overlapped Speech Model Parameters for Multichannel Speech Activity Detection in Meetings. *Proc. of ICASSP*, Toulouse, France, I:993–996.

T. Pfau and D. Ellis and A. Stolcke. 2001. Multispeaker Speech Activity Detection for the ICSI Meeting Recorder. *Proc. of ASRU*, Madonna di Campiglio, Italy, pp107–110.

S. Wrigley, G. Brown, V. Wan, and S. Renals. 2003. Feature Selection for the Classification of Crosstalk in Multi-Channel Audio. *Proc. of EUROSPEECH*, Geneva, Switzerland, pp469–472.

S. Wrigley, G. Brown, V. Wan, and S. Renals. 2005. Speech and Crosstalk Detection in Multichannel Audio. *IEEE Trans. on Speech and Audio Processing*, **13**:1, pp84–91.

Detection of Non-native Sentences using Machine-translated Training Data

John Lee
Spoken Language Systems
MIT CSAIL
Cambridge, MA 02139, USA
jsylee@csail.mit.edu

Ming Zhou, Xiaohua Liu
Natural Language Computing Group
Microsoft Research Asia
Beijing, 100080, China
{mingzhou,xiaoliu}@microsoft.com

Abstract

Training statistical models to detect non-native sentences requires a large corpus of non-native writing samples, which is often not readily available. This paper examines the extent to which machine-translated (MT) sentences can substitute as training data.

Two tasks are examined. For the native vs non-native *classification* task, non-native training data yields better performance; for the *ranking* task, however, models trained with a large, publicly available set of MT data perform as well as those trained with non-native data.

1 Introduction

For non-native speakers writing in a foreign language, feedback from native speakers is indispensable. While humans are likely to provide higher-quality feedback, a computer system can offer better availability and privacy. A system that can distinguish *non-native* ("ill-formed") English sentences from *native* ("well-formed") ones would provide valuable assistance in improving their writing.

Classifying a sentence into discrete categories can be difficult: a sentence that seems fluent to one judge might not be good enough to another. An alternative is to rank sentences by their relative fluency. This would be useful when a non-native speaker is unsure which one of several possible ways of writing a sentence is the best.

We therefore formulate two tasks on this problem. The **classification** task gives one sentence to the system, and asks whether it is native or non-native. The **ranking** task submits sentences with the same intended meaning, and asks which one is best.

To tackle these tasks, hand-crafting formal rules would be daunting. Statistical methods, however, require a large corpus of non-native writing samples, which can be difficult to compile. Since machine-translated (MT) sentences are readily available in abundance, we wish to address the question of whether they can substitute as training data.

The next section provides background on related research. Sections 3 and 4 describe our experiments, followed by conclusions and future directions.

2 Related Research

Previous research has paid little attention to ranking sentences by fluency. As for classification, one line of research in MT evaluation is to evaluate the fluency of an output sentence without its reference translations, such as in (Corston-Oliver et al., 2001) and (Gamon et al., 2005). Our task here is similar, but is applied on non-native sentences, arguably more challenging than MT output.

Evaluation of non-native writing has encompassed both the document and sentence levels. At the document level, automatic essay scorers, such as (Burstein et al., 2004) and (Ishioka and Kameda, 2006), can provide holistic scores that correlate well with those of human judges.

At the sentence level, which is the focus of this paper, previous work follows two trends. Some researchers explicitly focus on individual classes of er-

Proceedings of NAACL HLT 2007, Companion Volume, pages 93–96,
Rochester, NY, April 2007.

rors, e.g., mass vs count nouns in (Brockett et al., 2006) and (Nagata et al., 2006). Others implicitly do so with hand-crafted rules, via templates (Heidorn, 2000) or mal-rules in context-free grammars, such as (Michaud et al., 2000) and (Bender et al., 2004).

Typically, however, non-native writing exhibits a wide variety of errors, in grammar, style and word collocations. In this research, we allow unrestricted classes of errors[1], and in this regard our goal is closest to that of (Tomokiyo and Jones, 2001). However, they focus on non-native speech, and assume the availability of non-native training data.

3 Experimental Set-Up

3.1 Data

Our data consists of pairs of English sentences, one native and the other non-native, with the same "intended meaning". In our MT data (`MT`), both sentences are translated, by machine or human, from the same sentence in a foreign language. In our non-native data (`JLE`), the non-native sentence has been edited by a native speaker[2]. Table 1 gives some examples, and Table 2 presents some statistics.

`MT` (Multiple-Translation Chinese and Multiple-Translation Arabic corpora) English MT output, and human reference translations, of Chinese and Arabic newspaper articles.

`JLE` (Japanese Learners of English Corpus) Transcripts of Japanese examinees in the Standard Speaking Test. False starts and disfluencies were then cleaned up, and grammatical mistakes tagged (Izumi et al., 2003). The speaking style is more formal than spontaneous English, due to the examination setting.

3.2 Machine Learning Framework

SVM-Light (Joachims, 1999), an implementation of Support Vector Machines (SVM), is used for the classification task.

For the ranking task, we utilize the ranking mode of SVM-Light. In this mode, the SVM algorithm is adapted for learning ranking functions, originally used for ranking web pages with respect to a query (Joachims, 2002). In our context, given a set of English sentences with similar semantic content, say $s_1, \ldots, s_n$, and a ranking based on their fluency, the learning algorithm estimates the weights $\vec{w}$ to satisfy the inequalities:

$$\vec{w} \cdot \Phi(s_j) > \vec{w} \cdot \Phi(s_k) \quad (1)$$

where s_j is more fluent than s_k, and where Φ maps a sentence to a feature vector. This is in contrast to standard SVMs, which learn a hyperplane boundary between native and non-native sentences from the inequalities:

$$y_i(\vec{w} \cdot \Phi(s_i) + w_0) - 1 \geq 0 \quad (2)$$

where $y_i = \pm 1$ are the labels. Linear kernels are used in our experiments, and the regularization parameter is tuned on the development sets.

3.3 Features

The following features are extracted from each sentence. The first two are real numbers; the rest are indicator functions of the presence of the lexical and/or syntactic properties in question.

`Ent` Entropy[3] from a trigram language model trained on 4.4 million English sentences with the SRILM toolkit (Stolcke, 2002). The trigrams are intended to detect local mistakes.

`Parse` Parse score from Model 2 of the statistical parser (Collins, 1997), normalized by the number of words. We hypothesize that non-native sentences are more likely to receive lower scores.

`Deriv` Parse tree derivations, i.e., from each parent node to its children nodes, such as S → NP VP. Some non-native sentences have plausible N-grams, but have derivations infrequently seen in well-formed sentences, due to their unusual syntactic structures.

`DtNoun` Head word of a base noun phrase, and its determiner, e.g., (*the, markets*) from the human non-native sentence in Table 1. The usage of articles has been found to be the most frequent error class in the `JLE` corpus (Izumi et al., 2003).

[1] Except spelling mistakes, which we consider to be a separate problem that should be dealt with in a pre-processing step.

[2] The nature of the non-native data constrains the ranking to two sentences at a time.

[3] Entropy $H(x)$ is related to perplexity $PP(x)$ by the equation $PP(x) = 2^{H(x)}$.

Type		Sentence
Native	Human	New York and London stock markets went up
Non-native	Human	The stock markets in New York and London were increasing together
	MT	The same step of stock market of London of New York rises

Table 1: *Examples of sentences translated from a Chinese source sentence by a native speaker, by a non-native speaker, and by a machine translation system.*

Data Set	Corpus	# sentences (for classification)			# pairs (for ranking)
		total	native	non-native	
`MT train`	LDC{2002T01, 2003T18, 2006T04}	30075	17508	12567	91795
`MT dev`	LDC2003T17 (*Zaobao* only)	1995	1328	667	2668
`MT test`	LDC2003T17 (*Xinhua* only)	3255	2184	1071	4284
`JLE train`	Japanese Learners of English	9848	4924	4924	4924
`JLE dev`		1000	500	500	500
`JLE test`		1000	500	500	500

Table 2: *Data sets used in this paper.*

`Colloc` An in-house dependency parser extracts five types of word dependencies[4]: subject-verb, verb-object, adjective-noun, verb-adverb and preposition-object. For the human non-native sentence in Table 1, the unusual subject-verb collocation "*market increase*" is a useful clue in this otherwise well-formed sentence.

4 Analysis

4.1 An Upper Bound

To gauge the performance upper bound, we first attempt to classify and rank the MT test data, which should be less challenging than non-native data. After training the SVM on `MT train`, classification accuracy on `MT test` improves with the addition of each feature, culminating at 89.24% with all five features. This result compares favorably with the state-of-the-art[5]. Ranking performance reaches 96.73% with all five features.

We now turn our attention to non-native test data, and contrast the performance on `JLE test` using models trained by MT data (`MT train`), and by non-native data (`JLE train`).

[4]Proper nouns and numbers are replaced with special symbols. The words are further stemmed using Porter's Stemmer.

[5]Direct comparison is impossible since the corpora were different. (Corston-Oliver et al., 2001) reports 82.89% accuracy on English software manuals and online help documents, and (Gamon et al., 2005) reports 77.59% on French technical documents.

Test Set: `JLE test`	Train Set	
	`MT train`	`JLE train`
`Ent+ Parse`	57.2 (+) 48.6 (-) 65.8	57.7 (+) 70.6 (-) 44.8
`+Deriv`	58.4 (+) 54.6 (-) 62.2	64.7 (+)72.2 (-) 57.2
`+DtNoun`	**59.0** (+) 57.6 (-) 60.4	**66.4** (+) 72.8 (-) 60.0
`+Colloc`	58.6 (+) 54.2 (-) 63.2	65.9 (+) 72.6 (-) 59.2

Table 3: *Classification accuracy on `JLE test`. (-) indicates accuracy on non-native sentences, and (+) indicates accuracy on native sentences. The overall accuracy is their average.*

4.2 Classification

As shown in Table 3, classification accuracy on `JLE test` is higher with the `JLE train` set (66.4%) than with the larger `MT train` set (59.0%). The SVM trained on `MT train` consistently misclassifies more native sentences than non-native ones. One reason might be that speech transcripts have a less formal style than written news sentences. Transcripts of even good conversational English do not always resemble sentences in the news domain.

4.3 Ranking

In the ranking task, the relative performance between MT and non-native training data is reversed.

Test Set: JLE test	Train Set	
	MT train	JLE train
Ent+Parse	72.8	71.4
+Deriv	73.4	73.6
+DtNoun	75.4	73.8
+Colloc	**76.2**	**74.6**

Table 4: *Ranking accuracy on JLE test.*

As shown in Table 4, models trained on MT train yield higher ranking accuracy (76.2%) than those trained on JLE train (74.6%). This indicates that MT training data can generalize well enough to perform better than a non-native training corpus of size up to 10000.

The contrast between the classification and ranking results suggests that train/test data mismatch is less harmful for the latter task. Weights trained on the classification inequalities in (2) and on the ranking inequalities in (1) both try to separate native and MT sentences maximally. The absolute boundary learned in (2) is inherently specific to the nature of the training sentences, as we have seen in §4.2. In comparison, the relative scores learned from (1) have a better chance to carry over to other domains, as long as some gap still exists between the scores of the native and non-native sentences.

5 Conclusions & Future Work

We explored two tasks in sentence-level fluency evaluation: ranking and classifying native vs. non-native sentences. In an SVM framework, we examined how well MT data can replace non-native data in training.

For the classification task, training with MT data is less effective than with non-native data. However, for the ranking task, models trained on publicly available MT data generalize well, performing as well as those trained with a non-native corpus of size 10000.

In the future, we would like to search for more salient features through a careful study of non-native errors, using error-tagged corpora such as (Izumi et al., 2003). We also plan to explore techniques for combining large MT training corpora and smaller non-native training corpora. Our ultimate goal is to identify the errors in the non-native sentences and propose corrections.

References

E. Bender, D. Flickinger, S. Oepen, A. Walsh, and T. Baldwin. 2004. Arboretum: Using a Precision Grammar for Grammar Checking in CALL. *Proc. InSTIL/ICALL Symposium on Computer Assisted Learning*.

C. Brockett, W. Dolan, and M. Gamon. 2006. Correcting ESL Errors using Phrasal SMT Techniques. *Proc. ACL.*

J. Burstein, M. Chodorow and C. Leacock. 2004. Automated Essay Evaluation: The Criterion online Writing Service. *AI Magazine*, 25(3):27–36.

M. Collins. 1997. Three Generative, Lexicalised Models for Statistical Parsing. *Proc. ACL.*

S. Corston-Oliver, M. Gamon and C. Brockett. 2001. A Machine Learning Approach to the Automatic Evaluation of Machine Translation. *Proc. ACL.*

M. Gamon, A. Aue, and M. Smets. 2005. Sentence-Level MT Evaluation without Reference Translations: Beyond Language Modeling. *Proc. EAMT.*

G. Heidorn. 2000. Intelligent Writing Assistance. *Handbook of Natural Language Processing*. Robert Dale, Hermann Moisi and Harold Somers (ed.). Marcel Dekker, Inc.

T. Ishioka and M. Kameda. 2006. Automated Japanese Essay Scoring System based on Articles Written by Experts. *Proc. ACL.*

E. Izumi, K. Uchimoto, T. Saiga, T. Supnithi, and H. Isahara. 2003. Automatic Error Detection in the Japanese Learners' English Spoken Data. *Proc. ACL.*

T. Joachims. 1999. Making Large-Scale SVM Learning Practical. *Advances in Kernel Methods - Support Vector Learning*. B. Schölkopf, C. Burges and A. Smola (ed.), MIT-Press.

T. Joachims. 2002. Optimizing Search Engines using Clickthrough Data. *Proc. SIGKDD.*

L. Michaud, K. McCoy and C. Pennington. 2000. An Intelligent Tutoring System for Deaf Learners of Written English. *Proc. 4th International ACM Conference on Assistive Technologies*.

R. Nagata, A. Kawai, K. Morihiro, and N. Isu. 2006. A Feedback-Augmented Method for Detecting Errors in the Writing of Learners of English. *Proc. ACL.*

A. Stolcke. 2002. SRILM — An Extensible Language Modeling Toolkit *Proc. ICSLP.*

L. Tomokiyo and R. Jones. 2001. You're not from 'round here, are you? Naïve Bayes Detection of Non-native Utterance Text. *Proc. NAACL.*

Exploiting Rich Syntactic Information for Relation Extraction from Biomedical Articles*

Yudong Liu and **Zhongmin Shi** and **Anoop Sarkar**
School of Computing Science
Simon Fraser University
{yudongl,zshi1,anoop}@cs.sfu.ca

Abstract

This paper proposes a ternary relation extraction method primarily based on rich syntactic information. We identify PROTEIN-ORGANISM-LOCATION relations in the text of biomedical articles. Different kernel functions are used with an SVM learner to integrate two sources of information from syntactic parse trees: (i) a large number of syntactic features that have been shown useful for Semantic Role Labeling (SRL) and applied here to the relation extraction task, and (ii) features from the entire parse tree using a tree kernel. Our experiments show that the use of rich syntactic features significantly outperforms shallow word-based features. The best accuracy is obtained by combining SRL features with tree kernels.

1 Introduction

Biomedical functional relations (relations for short) state interactions among biomedical substances. For instance, the PROTEIN-ORGANISM-LOCATION (POL) relation that we study in this paper provides information about where a PROTEIN is located in an ORGANISM, giving a valuable clue to the biological function of the PROTEIN and helping to identify suitable drug, vaccine and diagnostic targets. Fig. 1 illustrates possible locations of proteins in Gram+ and Gram− bacteria. Previous work in biomedical relation extraction task (Sekimizu et al., 1998; Blaschke et al., 1999; Feldman et al., 2002) suggested the use of predicate-argument structure by taking verbs as the center of the relation – in contrast, in this paper we directly link protein named entities (NEs) to their locations; in other related work, (Claudio et al., 2006) proposed an approach that solely considers the shallow semantic features extracted from sentences.

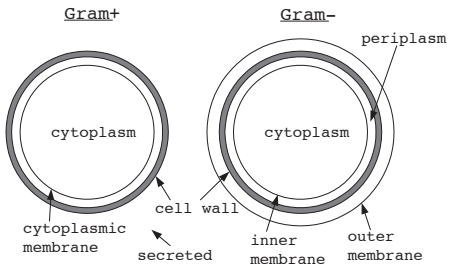


Figure 1: Illustration of bacterial locations

For relation extraction in the newswire domain, syntactic features have been used in a generative model (Miller et al., 2000) and in a discriminative log-linear model (Kambhatla, 2004). In comparison, we use a much larger set of syntactic features extracted from parse trees, many of which have been shown useful in SRL task. Kernel-based methods have also been used for relation extraction (Zelenko et al., 2003; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005) on various syntactic representations, such as dependency trees or constituency-based parse trees. In contrast, we explore a much wider variety of syntactic features in this work. To benefit from both views, a composite kernel (Zhang et al., 2006) integrates the flat features from entities and structured features from parse trees. In our work, we also combine a linear kernel with a tree kernel for improved performance.

2 SRL Features for Information Extraction

Fig. 2 shows one example illustrating the ternary relation we are identifying. In this example, "Exoenzyme S" is a PROTEIN name, "extracellular" a LOCATION name and "Pseudomonas aeruginosa" an ORGANISM name. Our task is to identify if there exists a "PROTEIN-ORGANISM-LOCATION" relation among these three NEs.

To simplify the problem, we first reduce the POL

*This research was partially supported by NSERC, Canada.

Proceedings of NAACL HLT 2007, Companion Volume, pages 97–100,
Rochester, NY, April 2007.

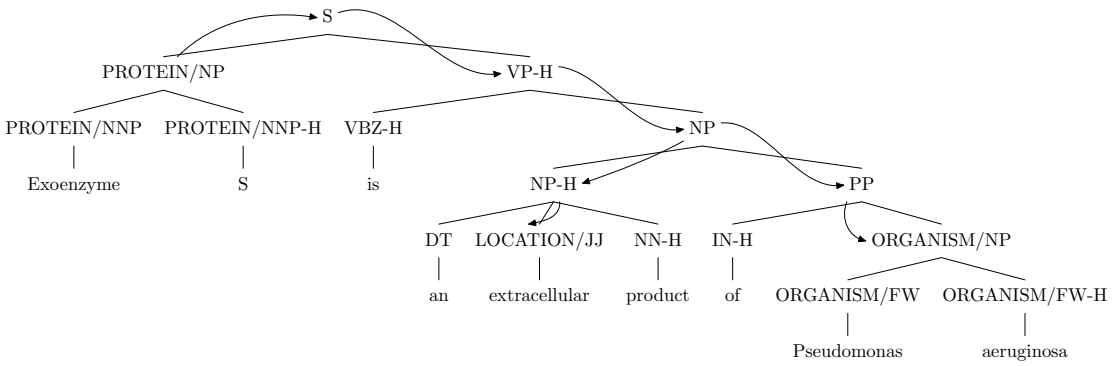


Figure 2: An example of POL ternary relation in a parse tree

ternary relation extraction problem into two binary relation extraction problems. Specifically, we split the POL ternary relation into binary relations as: (1) PO: PROTEIN and ORGANISM, and (2) PL: PROTEIN and LOCATION.

The ORGANISM-LOCATION relation is ignored because it does not consider the PROTEIN and is less meaningful than the PO and PL relations. Based on this simplification, and following the idea of SRL, we take the PROTEIN name in the role of the predicate (verb) and the ORGANISM/LOCATION name as its argument candidates in question. Then the problem of identifying the binary relations of PO and PL has been reduced to the problem of argument classification problem given the predicate and the argument candidates. The reason we pick PROTEIN names as predicates is that we assume PROTEIN names play a more central role in linking the binary relations to the final ternary relations.

Compared to a corpus for the standard SRL task, there are some differences in this task: first is the relative position of PROTEIN names and ORGANISM/LOCATION names. Unlike the case in SRL, where arguments locate either before or after the predicate, in this application it is possible that one NE is embedded in another. A second difference is that a predicate in SRL scenario typically consists of only one word; here a PROTEIN name can contain up to 8 words.

We do not use PropBank data in our model at all. All of our training data and test data is annotated by domain expert biologists and parsed by Charniak-Johnson's parser (released in 2006). When there is a misalignment between the NE and the constituent in the parse tree, we insert a new NP parent node for the NE.

3 System Description

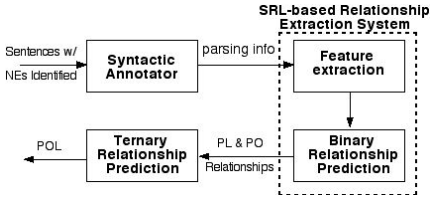


Figure 3: High-level system architecture

Fig. 3 shows the system overview. The input to our system consists of titles and abstracts that are extracted from MEDLINE records. These extracted sentences have been annotated with the NE information (PROTEIN, ORGANISM and LOCATION). The Syntactic Annotator parses the sentences and inserts the head information to the parse trees by using the Magerman/Collins head percolation rules. The main component of the system is our SRL-based relation extraction module, where we first manually extract features along the path from the PROTEIN name to the ORGANISM/LOCATION name and then train a binary SVM classifier for the binary relation extraction. Finally, we fuse the extracted binary relations into a ternary relation. In contrast with our discriminative model, a statistical parsing based generative model (Shi et al., 2007) has been proposed for a related task on this data set where the NEs and their relations are extracted together and used to identify which NEs are relevant in a particular sentence. Since our final goal is to facilitate the biologists to generate the annotated corpus, in future

• each word and its Part-of-Speech (POS) tag of PRO name
• head word (hw) and its POS of PRO name
• subcategorization that records the immediate structure that expands from PRO name. Non-PRO daughters will be eliminated
• POS of parent node of PRO name
• hw and its POS of the parent node of PRO name
• each word and its POS of ORG name (in the case of "PO " relation extraction).
• hw and its POS of ORG name
• POS of parent node of ORG name
• hw and its POS of the parent node of ORG name
• POS of the word immediately before/after ORG name
• punctuation immediately before/after ORG name
• feature combinations: hw of PRO name_hw of ORG name, hw of PRO name_POS of hw of ORG name, POS of hw of PRO name_POS of hw of ORG name
• path from PRO name to ORG name and the length of the path
• trigrams of the path. We consider up to 9 trigrams
• lowest common ancestor node of PRO name and ORG name along the path
• LCA (Lowest Common Ancestor) path that is from ORG name to its lowest common ancestor with PRO name
• relative position of PRO name and ORG name. In parse trees, we consider 4 types of positions that ORGs are relative to PROs: before, after, inside, other

Table 1: Features adopted from the SRL task. PRO: PROTEIN; ORG: ORGANISM

• subcategorization that records the immediate structure that expands from ORG name. Non-ORG daughters will be eliminated
• if there is an VP node along the path as ancestor of ORG name
• if there is an VP node as sibling of ORG name
• path from PRO name to LCA and the path length (L1)
• path from ORG name to LCA and the path length (L2)
• combination of L1 and L2
• sibling relation of PRO and ORG
• distance between PRO name and ORG name in the sentence. (3 valued: 0 if nw (number of words) = 0; 1 if $0 < nw <= 5$; 2 if $nw > 5$)
• combination of distance and sibling relation

Table 2: New features used in the SRL-based relation extraction system.

work we plan to take the relevant labeled NEs from the generative model as our input.

Table 1 and Table 2 list the features that are used in the system.

4 Experiments and Evaluation

4.1 Data set

Our experimental data set is derived from a small expert-curated corpus, where the POL relations and relevant PROTEIN, ORGANISM and LOCATION NEs are labeled. It contains ∼150k words, 565 relation instances for POL, 371 for PO and 431 for PL.

4.2 Systems and Experimental Results

We built several models to compare the relative utility of various types of rich syntactic features that we can exploit for this task. For various representations, such as feature vectors, trees and their combinations, we applied different kernels in a Support Vector Machine (SVM) learner. We use Joachims' SVM_light[1] with default linear kernel to feature vectors and Moschetti's SVM-light-TK-1.2[2] with the default tree kernel. The models are:

Baseline1 is a purely word-based system, where the features consist of the unigrams and bigrams between the PROTEIN name and the ORGANISM/LOCATION names inclusively, where the stopwords are selectively eliminated.

Baseline2 is a naive approach that assumes that any example containing PROTEIN, LOCATION names has the PL relation. The same assumption is made for PO and POL relations.

PAK system uses predicate-argument structure kernel (PAK) based method. PAK was defined in (Moschitti, 2004) and only considers the path from the *predicate* to the *target argument*, which in our setting is the path from the PROTEIN to the ORGANISM or LOCATION names.

SRL is an SRL system which is adapted to use our new feature set. A default linear kernel is applied with SVM learning.

TRK system is similar to PAK system except that the input is an entire parse tree instead of a PAK path.

TRK+SRL combines full parse trees and manually extracted features and uses the kernel combination.

[1] *http://svmlight.joachims.org/*

[2] *http://ai-nlp.info.uniroma2.it/moschitti/TK1.2-software/Tree-Kernel.htm*

Method	PL				PO				POL			
Measure	**Prec**	**Rec**	**F**	**Acc**	**Prec**	**Rec**	**F**	**Acc**	**Prec**	**Rec**	**F**	**Acc**
Baseline1	98.1	61.0	75.3	60.6	88.4	59.7	71.3	58.5	57.1	90.9	70.1	56.3
Baseline2	61.9	100.0	76.5	61.9	48.8	100.0	65.6	48.9	59.8	100.0	74.8	59.8
PAK	71.0	71.0	71.0	64.6	69.0	66.7	67.8	61.8	66.0	69.9	67.9	62.6
SRL	72.9	77.1	74.9	70.3	66.0	71.0	68.4	64.5	70.6	67.5	69.0	65.8
TRK	69.8	**81.6**	75.3	72.0	64.2	**84.1**	72.8	72.0	**79.6**	66.2	72.3	71.3
TRK+SRL	**74.9**	79.4	**77.1**	**72.8**	**73.9**	78.1	**75.9**	**72.6**	75.3	**74.5**	**74.9**	**71.8**

Table 3: Percent scores of Precision/Recall/F-score/Accuracy for identifying PL, PO and POL relations.

4.3 Fusion of Binary relations

We predict the POL ternary relation by fusing PL and PO binary relations if they belong to the same sentence and have the same PROTEIN NE. The prediction is made by the sum of confidence scores (produced by the SVM) of the PL and PO relations. This is similar to the postprocessing step in SRL task in which the semantic roles assigned to the arguments have to realize a legal final semantic frame for the given predicate.

4.4 Discussion

Table 3 shows the results using 5-fold cross validation. We report figures on ternary relation extraction and extraction of the two binary relations. Comparison between the **PAK** model and **SRL** model shows that manually specified features are more discriminative for binary relation extraction; they boost precision and accuracy for ternary relation extraction. In contrast to the **SRL** model for binary relation extraction, the **TRK** model obtains lower recall but higher precision. The combination of **SRL** with the **TRK** system gives best overall accuracy of 71.8% outperforming shallow word based features.

5 Conclusion

In this paper we explored the use of rich syntactic features for the relation extraction task. In contrast with the previously used set of syntactic features for this task, we use a large number of features originally proposed for the Semantic Role Labeling task. We provide comprehensive experiments using many different models that use features from parse trees. Using rich syntactic features by combining SRL features with tree kernels over the entire tree obtains 71.8% accuracy which significantly outperforms shallow word-based features which obtains 56.3% accuracy.

References

C. Blaschke, M. Andrade, C. Ouzounis, and A. Valencia. 1999. Automatic extraction of biological information from scientific text: Protein-protein interactions. In *AAAI-ISMB 1999*.

R. C. Bunescu and R. J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proc. HLT/EMNLP-2005*.

G. Claudio, A. Lavelli, and L. Romano. 2006. Exploiting Shallow Linguistic Information for Relation Extraction from Biomedical Literature. In *Proc. EACL 2006*.

A. Culotta and J. Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proc. ACL-2004*.

R. Feldman, Y. Regev, M. Finkelstein-Landau, E. Hurvitz, and B. Kogan. 2002. Mining biomedical literature using information extraction. *Current Drug Discovery*.

N. Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction. In *Proc. ACL-2004 (poster session)*.

S. Miller, H. Fox, L. Ramshaw, and R. Weischedel. 2000. A novel use of statistical parsing to extract information from text. *Proc. NAACL-2000*.

A. Moschitti. 2004. A study on convolution kernels for shallow semantic parsing. In *Proc. ACL-2004*.

T. Sekimizu, H.S. Park, and J. Tsujii. 1998. Identifying the interaction between genes and gene products based on frequently seen verbs in medline abstracts. In *Genome Informatics*. 62-71.

Z. Shi, A. Sarkar, and F. Popowich. 2007. Simultaneous Identification of Biomedical Named-Entity and Functional Relation UsingStatistical Parsing Techniques. In *NAACL-HLT 2007 (short paper)*.

D. Zelenko, C. Aone, and A. Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*.

M. Zhang, J. Zhang, J. Su, and G.D. Zhou. 2006. A Composite Kernel to Extract Relations between Entities with Both Flat and Structured Features. In *Proc. ACL-2006*.

Look Who is Talking: Soundbite Speaker Name Recognition in Broadcast News Speech

Feifan Liu, Yang Liu
Department of Computer Science
The University of Texas at Dallas, Richardson, TX
{ffliu,yangl}@hlt.utdallas.edu

Abstract

Speaker name recognition plays an important role in many spoken language applications, such as rich transcription, information extraction, question answering, and opinion mining. In this paper, we developed an SVM-based classification framework to determine the speaker names for those included speech segments in broadcast news speech, called soundbites. We evaluated a variety of features with different feature selection strategies. Experiments on Mandarin broadcast news speech show that using our proposed approach, the soundbite speaker name recognition (SSNR) accuracy is 68.9% on our blind test set, an absolute 10% improvement compared to a baseline system, which chooses the person name closest to the soundbite.

1 Introduction

Broadcast news (BN) speech often contains speech or interview quotations from specific speakers other than reporters and anchors in a show. Identifying speaker names for these speech segmentations, called soundbites (Maskey and Hirschberg, 2006), is useful for many speech processing applications, e.g., question answering, opinion mining for a specific person. This has recently received increasing attention in programs such as the DARPA GALE program, where one query template is about a person's opinion or statement.

Previous work in this line includes speaker role detection (e.g., Liu, 2006; Maskey and Hirschberg, 2006) and speaker diarization (e.g., Canseco et al., 2005). In this paper, we formulate the problem of SSNR as a traditional classification task, and proposed an SVM-based identification framework to explore rich linguistic features. Experiments on Mandarin BN speech have shown that our proposed approach significantly outperforms the baseline system, which chooses the closest name as the speaker for a soundbite.

2 Related Work

To our knowledge, no research has yet been conducted on soundbite speaker name identification in Mandarin BN domain. However, this work is related to some extent to speaker role identification, speaker diarization, and named entity recognition.

Speaker role identification attempts to classify speech segments based on the speakers' role (anchor, reporter, or others). Barzilay et al. (2000) used BoosTexter and the maximum entropy model for this task in English BN corpus, obtaining a classification accuracy of about 80% compared to the chance of 35%. Liu (2006) combined a generative HMM approach with the conditional maximum entropy method to detect speaker roles in Mandarin BN, reporting a classification accuracy of 81.97% against the baseline of around 50%. In Maskey and Hirschberg (2006), the task is to recognize soundbites (which make up of a large portion of the "other" role category in Liu (2006)). They achieved a recognition accuracy of 67.4% in the English BN domain. Different from their work, our goal is to identify the person who spoke those soundbites, i.e., associate each soundbite with a speaker name if any.

Speaker diarization in BN aims to find speaker changes, group the same speakers together, and recognize speaker names. It is an important component for rich transcription (e.g., in the DARPA EARS program). So far most work in this area has only focused on speaker segmentation and clustering, and not included name recognition. However, Canseco et al. (2005) were able to successfully use linguistic information (e.g., related to person names) to improve performance of BN speaker segmentation and clustering.

This work is also related to named entity recognition (NER), especially person names. There has been a large amount of research efforts on NER; however, instead of

Proceedings of NAACL HLT 2007, Companion Volume, pages 101–104,
Rochester, NY, April 2007.

recognizing all the names in a document, our task is to find the speaker for a particular speech segment.

3 Framework for Soundbite Speaker Name Recognition (SSNR)

Figure 1 shows our system diagram. SSNR is conducted using the speech transcripts, assuming the soundbite segments are provided. After running NER in the transcripts, we obtain candidate person names. For a soundbite, we use the name hypotheses from the region both before and after the soundbite. A 'region' is defined based on the turn and topic segmentation information. To determine which name among the candidates is the corresponding speaker for the soundbite, we recast this problem as a binary classification problem for every candidate name and the soundbite, which we call an instance. A positive tag for an instance means that the name is the soundbite speaker. Each instance has an associated feature vector, described further in the following section. Note that if a name occurs more than once, only one instance is created for it.

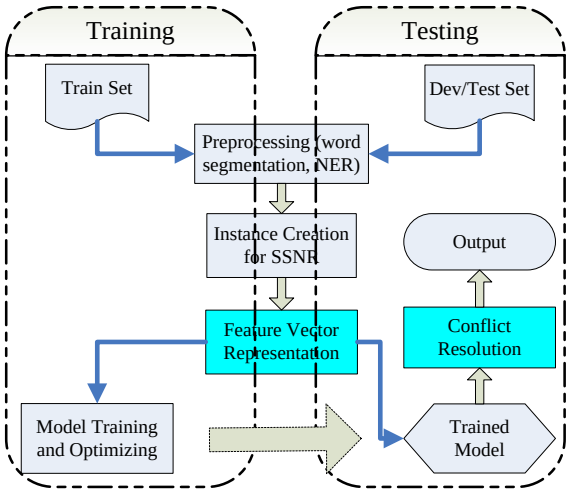


Figure 1. System diagram for SSNR.

Any classification approach can be used in this general framework for SSNR. We choose to use an SVM classifier in our experiments because of its superior performance in many classification tasks.

3.1 Features

The features that we have explored can be grouped into three categories.

Positional Features (PF)

- PF-1: the position of the candidate name relative to the soundbite. We hypothesize that names closer to a soundbite are more likely to be the soundbite speaker. This feature value can be 'last', 'first', 'mid', or 'unique'. For example, 'last' for a candidate before a soundbite means that it is the closest name among the hypotheses before the soundbite. 'Unique' indicates that the candidate is the only person name in the region before or after the soundbite. Note that if a candidate name occurs more than once, the PF-1 feature corresponds to the closest name to the soundbite.
- PF-2: the position of a name in its sentence. Typically a name appearing earlier in a sentence (e.g., a subject) is more likely to be quoted later.
- PF-3: an indicator feature to show where the name has occurred, before, inside, or after the soundbite. We added this because it is rare that a name inside a soundbite is the speaker of that soundbite.
- PF-4: an indicator to denote if a candidate is in the last sentence just before the soundbite turn, or is in the first sentence just after the soundbite turn.

Frequency Features (Freq)

We hypothesize that a name with more occurrences might be an important subject and thus more likely to be the speaker of the soundbite, therefore we include the frequency of a candidate name in the feature set.

Lexical Features (LF)

In order to capture the cue words around the soundbite speaker names in the transcripts, we included unigram features. For example, "pre_word+1=说/said" denotes that the candidate name is followed by the word '说/said', and that 'pre' means this happens in the region before the soundbite.

3.2 Conflict Resolution

Another component in the system diagram that is worth pointing out is 'conflict resolution'. Since our approach treats each candidate name as a separate classification task, we need to post-process the cases where there are multiple or no positive hypotheses for a soundbite during testing. To resolve this situation, we choose the instance with the best confidence value from the classifier.

4 Experiments

4.1 Experimental Setup

We use the TDT4 Mandarin broadcast news data in our experiment. The data set consists of about 170 hours (336 shows) of news speech from different sources. Speaker turns and soundbite segment information were annotated manually in the transcripts. Our current study

only uses the soundbites that have a human-labeled speaker name in the surrounding transcripts. There are 1292 such soundbites in our corpus. We put aside 1/10 of the data as the development set, another 1/10 as the test set, and used the rest as our training set. All the transcripts were automatically tagged with named entities using the NYU tagger (Ji and Grishman, 2005). For the classifier, we used the libSVM toolkit (Chang and Lin, 2001) and the RBF kernel in our experiments.

A reasonable baseline for SSNR is to choose the closest person name before a soundbite as its speaker. We will compare our system performance to this baseline approach.

We used two performance metrics in our experiments. First is the instance classification accuracy (*CA*) for the candidate names in the framework of the binary classification task. Second, we compute name recognition accuracy (*RA*) for the soundbites as follows:

$$RA = \frac{\#\ of\ Soundbites\ with\ Correct\ Names}{\#\ of\ Soundbites\ in\ Files}$$

4.2 Effects of Different Manually Selected Feature Subsets

We used 10-fold cross validation on the training set to evaluate the effect of different features and also for parameter optimization. Table 1 shows the instance classification results. "PF, Freq, LF" are the features described in Section 3.1. "LF-before" means the unigram features before the soundbites. "All-before" denotes using all the features before the soundbites.

Feature Subsets	Optimized Para.		*CA* (%)
	C	G	
PF-1	0.125	2	83.48
+PF-2	2048	1.22e-4	85.62
+PF-3	2048	4.88e-4	85.79
+PF-4	2	0.5	86.18
+Freq	2	0.5	86.18
+LF-before	32	7.81e-3	88.44
+LF-after i.e., All features	8	0.0313	88.44
All-before	8	0.0313	88.03

Table 1. Instance classification accuracy (*CA)* using different feature sets. C and G are the optimized parameters in the SVM model.

We notice that the system performance generally improves with incrementally expended feature sets, yielding an accuracy of 88.44% using all the features. Some features seem not helpful to system performance, such as "Freq" and "LF-after". Using all the features before the soundbites achieves comparable performance to using all the features, indicating that the region before a soundbite contributes more than that after it. This is expected since the reporters typically have already mentioned the person's name before a soundbite. In addition, we evaluated some compound features using our current feature definition, but adding those did not improve the system performance.

4.3 Automatic Feature Selection

We also performed automatic feature selection for the SVM model based on the F-score criterion (Chen and Lin, 2006). There are 6048 features in total in our system. Figure 2 shows the classification performance in the training set using different number of features via automatic feature selection.

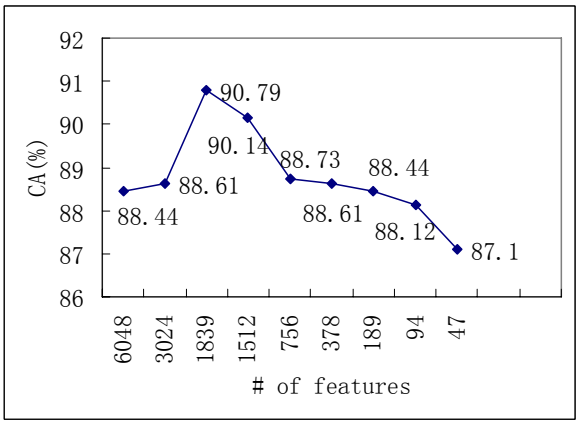


Figure 2. Instance classification accuracy (*CA*) using F-score based feature selection.

We can see that automatic feature selection further improves the classification performance (2.36% higher accuracy than that in Table 1). Table 2 lists some of the top features based on their F-scores. Consistent with our expectation, we observe that position related features, as well as cue words, are good indicators for SSNR.

Feature	F-score
Justbeforeturn (PF-4)	0.3543
pre_contextpos=last (PF-1)	0.2857
pre_senpos=unique (PF-2)	0.0631
pre_word+1="上午/morning" (LF)	0.0475
pre_word+1= "说/said" (LF)	0.0399
bool_pre=1 (PF-3)	0.0353
Justafterturn (PF-4)	0.0349
pre_contextpos=mid (PF-1)	0.0329
post_contextpos=first (PF-1)	0.0323
pre_word+1= "今天/today" (LF)	0.0288
pre_word-1="记者/reporter" (LF)	0.0251
pre_word+1="表示/express" (LF)	0.0246

Table 2. Top features ordered by F-score values.

4.4 Performance on Development Set

Up to now our focus has been on feature selection based on instance classification accuracy. Since our ultimate goal is to identify soundbite speaker names, we chose several promising configurations based on the results above to apply to the development set and evaluate the soundbite name recognition accuracy. Results using the two metrics are presented in Table 3.

Feature Set	*CA* (%)	*RA* (%)
Baseline	84.0	59.3
PF	86.7	54.2
PF+Freq	86.7	60.4
PF+Freq+LF-before	87.8	63.5
PF+Freq+LF-before +LF-after (ALL)	88.3	67.7
Top 1512 by f-score	85.6	62.5
Top 1839 by f-score	85.4	60.4

Table 3. Results on the dev set using two metrics: instance classification accuracy (*CA*), and soundbite name recognition accuracy (*RA*). The oracle *RA* is 79.1%.

Table 3 shows that using all the features (ALL) performs the best, yielding an improvement of 4.3% and 8.4% compared to the baseline in term of the *CA* and *RA* respectively. However, using the automatically selected feature sets (the last two rows in Table 3) only slightly outperforms the baseline. This suggests that the F-score based feature selection strategy on the training set may not generalize well. Interestingly, "Freq" and "LF-after" features show some useful contribution (the 4th and 6th row in Table 3) respectively on the development set, different from the results on the training set using 10-fold cross validation. The results using the two metrics also show that they are not always correlated.

Because of the possible NER errors, we also measure the oracle *RA*, defined as the percent of the soundbites for which the correct speaker name (based on NER) appears in the region surrounding the soundbite. The oracle *RA* on this data set is 79.1%. We also notice that 8.3% of the soundbites do not have the correct name hypothesis due to an NER boundary error, and that 12.5% is because of missing errors.

We used the method as described in Section 3.2 to resolve conflicts for the results shown in Table 3. In addition, we evaluated another approach—we resort to the baseline (i.e., chose the name that is closest to the soundbite) for those soundbites that have multiple or no positive hypothesis. Our experiments on the development set showed this approach degrades system performance (e.g., *RA* of around 61% using all the features).

4.5 Results on Blind Test Set

Finally, we applied the all-feature configuration to our blind test data and obtained the results as shown in Table 4. Using all the features significantly outperforms the baseline. The gain is slightly better than that on the development set, although the oracle accuracy is also higher on the test set.

	CA (%)	*RA* (oracle: 85.8%)
Baseline	81.3	58.4
All feature	85.1	68.9

Table 4. Results on the test set.

5 Conclusion

We proposed an SVM-based approach for soundbite speaker name recognition and examined various linguistic features. Experiments in Mandarin BN corpus show that our approach yields an identification accuracy of 68.9%, significantly better than 58.4% from the baseline.

Our future work will focus on exploring more useful features, such as part-of-speech and semantic features. In addition, we plan to test this framework using automatic speech recognition output, speaker segmentation, and soundbite segment detection.

6 Acknowledgement

We thank Sameer Maskey, Julia Hirschberg, and Mari Ostendorf for useful discussions, and Heng Ji for sharing the Mandarin named entity tagger. This work is supported by DARPA under Contract No. HR0011-06-C-0023. Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

References

S. Maskey and J. Hirschberg. 2006. Soundbite Detection in Broadcast News Domain. In *Proc. of INTERSPEECH2006*. pp: 1543-1546.

Y. Liu. 2006. Initial Study on Automatic Identification of Speaker Role in Broadcast News Speech. In *Proc. of HLT-NAACL*. pp: 81-84.

R. Barzilay, M. Collins, J. Hirschberg, and S. Whittaker. 2000. The Rules Behind Roles: Identifying Speaker Role in Radio Broadcasts. In *Proc. of AAAI*.

L. Canseco, L. Lamel, and J.-L. Gauvain. 2005. A Comparative Study Using Manual and Automatic Transcriptions for Diarization. In *Proc. of ASRU*.

H. Ji and R. Grishman. 2005. Improving Name Tagging by Reference Resolution and Relation Detection. In *Proc. of ACL*. pp: 411-418.

Y.-W. Chen and C.-J. Lin. 2006. Combining SVMs with Various Feature Selection Strategies. *Feature Extraction, Foundations and Applications*, Springer.

C. Chang and C. Lin. 2001. LIBSVM: A Library for Support Vector Machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Tagging Icelandic text using a linguistic and a statistical tagger

Hrafn Loftsson*
Department of Computer Science
Reykjavik University
Reykjavik IS-103, Iceland
hrafn@ru.is

Abstract

We describe our linguistic rule-based tagger *IceTagger*, and compare its tagging accuracy to the *TnT* tagger, a state-of-the-art statistical tagger, when tagging Icelandic, a morphologically complex language. Evaluation shows that the average tagging accuracy is 91.54% and 90.44%, obtained by *IceTagger* and *TnT*, respectively. When *tag profile gaps* in the lexicon, used by the *TnT* tagger, are filled with tags produced by our morphological analyser *IceMorphy*, *TnT's* tagging accuracy increases to 91.18%.

1 Introduction

In this paper, we use a *linguistic rule-based method* (LRBM) and a *data-driven method* (DDM) for tagging text in the morphologically complex Icelandic language.

We present a novel LRBM. The tagger based on this method, hereafter called *IceTagger*, uses about 175 local rules for initial disambiguation, and a set of heuristics, to force feature agreement where appropriate, for further disambiguation.

The average tagging accuracy of *IceTagger* is 91.54%, compared to 90.44% achieved by the *TnT* tagger, a state-of-the-art statistical tagger (Brants, 2000). *IceTagger* makes 11.5% less errors than *TnT*. On the other hand, when *tag profile gaps* in the lexicon, used by *TnT*, are filled with tags produced by *IceMorphy*, our morphological analyser, *TnT's* tagging accuracy increases to 91.18%. In that case, *IceTagger* makes 4.1% less errors than *TnT*.

The remainder of this paper is organised as follows: In Sect. 2, we describe the different tagging methods in more detail. Sect. 3 briefly describes the Icelandic language and the tagset. The components of *IceTagger* are described in Sect. 4, and evaluation results are presented in Sect. 5.

2 The tagging methods

DDMs use machine learning to automatically derive a language model from, usually, hand-annotated corpora. An advantage of the DDMs is their language and tagset independence property. Their disadvantage is that a tagged corpus is essential for training. Furthermore, the limited window size used for disambiguation (e.g. three words) can be responsible for some of the tagging errors.

One of the better known statistical data-driven tagger is the *TnT* tagger (written in C). The tagger uses a second order (trigram) Hidden Markov model. The probabilities of the model are estimated from a training corpus using maximum likelihood estimation. New assignments of part-of-speech (POS) to words is found by optimising the product of lexical probabilities ($p(w_i|t_j)$) and contextual probabilities ($p(t_i|t_{i-1}, t_{i-2})$) (where w_i and t_i are the ith word and tag, respectively).

In contrast to DDMs, LRBMs are developed with the purpose of tagging a specific language using a particular tagset. The purpose of the rules is, usually, to remove illegitimate tags from words based on context. The advantage of LRBMs is that they do not

* The author is also affiliated with the Dept. of Computer Science, University of Sheffield, Sheffield, S1 4DP, UK.

Proceedings of NAACL HLT 2007, Companion Volume, pages 105–108,
Rochester, NY, April 2007.

rely (to the same extent as DDMs) on the existence of a tagged corpus, and rules can be written to refer to words and tags in the entire sentence. The construction of a linguistic rule-based tagger, however, has been considered a difficult and time-consuming task (Voutilainen, 1995).

One of the better known LRBMs is the Constraint Grammar (CG) framework (Karlsson et al., 1995), in which both POS and grammatical functions are tagged. The EngCG-2 tagger, developed over several years and consisting of 3,600 rules, has been shown to obtain high accuracy (Samuelsson and Voutilainen, 1997).

The development time of our LRBM (written in Java; including a tokeniser, *IceMorphy* and *IceTagger*) was 7 man-months, which can be considered a short development time for a LRBM. This is mainly due to the emphasis on using heuristics (see Sect. 4.3) for disambiguation, as opposed to writing a large number of local rules.

3 The Icelandic language and its tagset

The Icelandic language is one of the Nordic languages. The language is morphologically rich, mainly due to inflectional complexity. A thorough description of the language can, for example, be found in (Þráinsson, 1994).

The main Icelandic tagset, constructed in the compilation of the tagged corpus *Icelandic Frequency Dictionary* (*IFD*) (Pind et al., 1991), is large (about 660 tags) compared to related languages. In this tagset, each character in a tag has a particular function. Table 1 shows the semantics of the noun and the adjective tags.

To illustrate, consider the phrase "*fallegu hestarnir*" (beautiful horses). The corresponding tag for "*fallegu*" is "*lkfnvf*", denoting adjective, masculine, plural, nominative, weak declension, positive; and the tag for "*hestarnir*" is "*nkfng*" denoting noun, masculine, plural, nominative with suffixed definite article.

4 IceTagger

IceTagger consists of three main components: an unknown word guesser, local rules for initial disambiguation and heuristics for further disambiguation. Both the local rules and the heuristics have been developed using linguistic knowledge and tuned using a development corpus (described in Sect. 5).

Char #	Category/ Feature	Symbol – semantics
1	Word class	**n**–noun, **l**–adjective
2	Gender	**k**–masculine, **v**–feminine, **h**–neuter, **x**–unspecified
3	Number	**e**–singular, **f**–plural,
4	Case	**n**–nominative, **o**–accusative, **þ**–dative, **e**–genitive
5	Article	**g**–with suffixed article
5	Declension	**s**–strong, **v**–weak
6	Proper noun	**m**–person, **ö**–place, **s**–other
6	Degree	**f**–positive, **m**–comparative, **e**–superlative

Table 1: The semantics of the noun and the adjective tags.

4.1 The unknown word guesser

The purpose of our morphological analyser *IceMorphy*, which is used as an unknown word guesser by *IceTagger*, is to generate all appropriate tags for a given word. It uses a familiar approach to unknown word guessing, i.e. it performs morphological/compound analysis and ending analysis (Mikheev, 1997; Nakov et al., 2003). Additionally, *IceMorphy* includes an important module for handling *tag profile gaps* (for a thorough description of *IceMorphy*, consult (Loftsson, 2006a)).

A *tag profile gap* arises when a particular word, listed in a lexicon derived from a corpus, has some missing tags in its tag profile (set of possible tags). The missing tag(s) might just not have been encountered during the derivation of the lexicon (e.g. during training). For each noun, adjective or verb, of a particular morphological class, *IceMorphy* is able to fill in the gaps for the given word.

To illustrate, consider the word "*konu*" (woman), and let us assume that only the tag "*nveo*" (denoting noun, feminine, singular, accusative) is found in the lexicon. Based on the "*u*" morphological suffix and the accusative case of the tag, *IceMorphy* assumes the word belongs to a particular morphological feminine noun class, in which singular accusative, dative and genitive cases have the same word form. Con-

sequently, IceMorphy generates the correct missing tags: "*nveþ*" and "*nvee*".

4.2 Local rules

The purpose of a local rule is to eliminate inappropriate tags from words, based on a local context (a window of 5 words; two words to the left and right of the focus word). This reductionistic approach is common in rule-based taggers. It is, for example, used in the CG systems.

In principle, the local rules are unordered. The firing of a rule is, however, dependent on the order of the words in a sentence. A sentence to be tagged is scanned from left to right and all tags of each word are checked in a sequence. Depending on the word class (the first letter of the tag) of the focus word, the token is sent to the appropriate disambiguation routine, which checks a variety of disambiguation constraints applicable to the particular word class and the surrounding words. At each step, only tags for the focus word are eliminated.

The format of a local rule is: *If* <*condition*> *eliminate tag t*. A <*condition*> is a boolean expression, whose individual components can refer to lexical forms or individual characters (word class/morphological features) of tags. The following are examples of <*condition*> (L_1/R_1 and L_2/R_2 denote tokens one and two to the left/right of the focus word, F, respectively):

L_1.isOnlyWordClass(x) AND L_2.isOnlyWordClass(y)
R_1.isWordClass(x) OR R_2.isWordClass(y)
L_1.isWordClass(x) AND t.isCase(y) AND t.isGender(z)
R_1.lexeme.equals(x) AND F.isWordClass(y)

To exemplify, consider the sentence part: "*við vorum* ..." (we were ...). The word "*við*" can have the following five tags ("_" is used as a separator between tags): "*ao_aþ_fp1fn_aa_nkeo*". For illustration purposes, it is sufficient to point out that the first two tags denote prepositions governing the accusative and the dative cases, respectively. Since the following word is a verb ("*vorum*") and prepositions only precede nominals, a rule, with <condition>=R_1.isOnlyWordClass(Verb), eliminates preposition tags in this context, leaving only the tags "*fp1fn_aa_nkeo*".

The current version of our tagger has 175 local rules. The rules are written in a separate file and compiled to Java code.

4.3 The heuristics

Once local disambiguation has been carried out, each sentence is sent to a global heuristic module, consisting of a collection of algorithmic procedures. Its purpose is to perform grammatical function analysis, guess prepositional phrases (PPs) and use the acquired knowledge to force feature agreement where appropriate. We call these heuristics global because, when disambiguating a particular word, a heuristic can refer to a word which is not in the nearest neighbourhood.

The heuristics repeatedly scan each sentence and perform the following: 1) *mark PPs*, 2) *mark verbs*, 3) *mark subjects*, 4) *force subject-verb agreement*, 5) *mark objects*, 6) *force subject-object agreement*, 7) *force verb-object agreement*, 8) *force nominal agreement*, and 9) *force PP agreement*. Lastly, the default heuristic is simply to choose the most frequent tag according to frequency information derived from the *IFD* corpus. A detailed description of all the heuristics can be found in (Loftsson, 2006b).

5 Evaluation

For evaluation, we used the pairs of ten training and test corpora of the *IFD* corpus, produced by Helgadóttir (2004). We used the first nine of these test corpora for evaluation, but the tenth one was set aside and used as the development corpus for *IceTagger*.

For each test corpus (10% of the *IFD*) the corresponding training corpus (90% of the *IFD*) was used to deduce the lexicon(s) used by *TnT*, *IceTagger* and *IceMorphy*. When testing the two taggers, we thus made sure that the ratio of unknown words was (almost) the same.

The accuracy of a base tagger, which assigns each known word its most frequent tag, and the most frequent noun tag/proper noun tag to lower case/upper case unknown words, is 76.27% (see table 2).

The average tagging accuracy of *IceTagger* for all words is 91.54%, compared to 90.44% for *TnT* (see table 2). *IceTagger* makes 11.5% less errors than *TnT*[1].

In order to improve the tagging accuracy of *TnT*, we used the *tag profile gap* filling mechanism of *Ice-*

[1] *TnT* is very fast, it tags about 50,000 tokens/sec on a Dell Optiplex GX620 Pentium 4, 3.20 GHz. *IceTagger* tags about 2,700 tokens/sec.

Words	Base	TnT	TnT*	IceTagger
Unkn.	4.39%	71.68%	72.75%	75.09%
Known	81.84%	91.82%	92.53%	92.74%
All	76.27%	90.44%	91.18%	91.54%

Table 2: Average tagging accuracy of the various taggers.

Morphy in the following manner. Each record in the lexicon used by *TnT* consists of a word and the corresponding tags found in the training corpus. Additionally, to facilitate lexical probability calculations, each tag is marked by its frequency (i.e. how often the tag appeared as a label for the given word). We made *IceMorphy* generate a "filled" lexicon such that each generated missing tag was marked with the frequency 1[2]. We call the resulting tagger *TnT**. Indeed, when testing *TnT**, we obtained an overall average tagging accuracy of 91.18% (see table 2). *IceTagger* makes 4.1% less errors than *TnT**.

The development of *IceTagger/IceMorphy* took 7 man-months, but it has been worth the effort. First, *IceTagger* does make fewer errors than *TnT*, and its accuracy can probably be increased by improving its individual components. Secondly, we have used *IceTagger* in various tagger combination methods to further increase the tagging accuracy of Icelandic text (Loftsson, 2006c).

6 Conclusion

In this paper, we have compared the tagging accuracy of our linguistic rule-based tagger, *IceTagger*, to the accuracy of *TnT*, a state-of-the-art statistical tagger.

IceTagger uses only about 175 local rules, but is able to achieve high accuracy through the use of global heuristics along with automatic *tag profile gap* filling. The average tagging accuracy of *IceTagger* is 91.54%, compared to 90.44% obtained by the *TnT* tagger. On the other hand, we were able to obtain 91.18% accuracy using *TnT* along with the *tag profile gap* filling mechanism of *IceMorphy*.

In future work, we would like to improve individual components of *IceTagger* and *IceMorphy*, with the purpose of further increasing the tagging accuracy.

[2]This seems logical since the missing tags were not found in the training corpus and are, hence, infrequent.

References

T. Brants. 2000. TnT: A statistical part-of-speech tagger. In *Proceedings of the* 6th *Conference on Applied natural language processing*, Seattle, WA, USA.

S. Helgadóttir. 2004. Testing Data-Driven Learning Algorithms for PoS Tagging of Icelandic. In H. Holmboe, editor, *Nordisk Sprogteknologi 2004*. Museum Tusculanums Forlag.

F. Karlsson, A. Voutilainen, J. Heikkilä, and A. Anttila. 1995. *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Mouton de Gruyter, Berlin, Germany.

H. Loftsson. 2006a. Tagging Icelandic text: A linguistic rule-based approach. Technical Report CS-06-04, Department of Computer Science, University of Sheffield.

H. Loftsson. 2006b. Tagging a Morphologically Complex Language Using Heuristics. In T. Salakoski, F. Ginter, S. Pyysalo, and T. Pahikkala, editors, *Advances in Natural Language Processing,* 5th *International Conference on NLP, FinTAL 2006, Proceedings*, Turku, Finland.

H. Loftsson. 2006c. Tagging Icelandic text: An experiment with integrations and combinations of taggers. *Language Resources and Evaluation*, 40(2):175–181.

A. Mikheev. 1997. Automatic Rule Induction for Unknown Word Guessing. *Computational Linguistics*, 21(4):543–565.

P. Nakov, Y. Bonev, G. Angelova, E. Cius, and W. Hahn. 2003. Guessing Morphological Classes of Unknown German Nouns. In *Proceedings of Recent Advances in Natural Language Processing*, Borovets, Bulgaria.

J. Pind, F. Magnússon, and S. Briem. 1991. *The Icelandic Frequency Dictionary*. The Institute of Lexicography, University of Iceland, Reykjavik, Iceland.

H. Þráinsson. 1994. Icelandic. In E. König and J. Auwera, editors, *The Germanic Languages*. Routledge, London.

C. Samuelsson and A Voutilainen. 1997. Comparing a Linguistic and a Stochastic tagger. In *Proceedings of the* 8th *Conference of the European Chapter of the ACL (EACL)*, Madrid, Spain.

A. Voutilainen. 1995. A syntax-based part-of-speech analyzer. In *Proceedings of the* 7th *Conference of the European Chapter of the ACL (EACL)*, Dublin, Ireland.

Efficient Computation of Entropy Gradient for Semi-Supervised Conditional Random Fields

Gideon S. Mann and **Andrew McCallum**
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
gideon.mann@gmail.com, mccallum@cs.umass.edu

Abstract

Entropy regularization is a straightforward and successful method of semi-supervised learning that augments the traditional conditional likelihood objective function with an additional term that aims to minimize the predicted label entropy on unlabeled data. It has previously been demonstrated to provide positive results in linear-chain CRFs, but the published method for calculating the entropy gradient requires significantly more computation than supervised CRF training. This paper presents a new derivation and dynamic program for calculating the entropy gradient that is significantly more efficient—having the same asymptotic time complexity as supervised CRF training. We also present efficient generalizations of this method for calculating the label entropy of all sub-sequences, which is useful for active learning, among other applications.

1 Introduction

Semi-supervised learning is of growing importance in machine learning and NLP (Zhu, 2005). Conditional random fields (CRFs) (Lafferty et al., 2001) are an appealing target for semi-supervised learning because they achieve state-of-the-art performance across a broad spectrum of sequence labeling tasks, and yet, like many other machine learning methods, training them by supervised learning typically requires large annotated data sets.

Entropy regularization (ER) is a method of semi-supervised learning first proposed for classification tasks (Grandvalet and Bengio, 2004). In addition to maximizing conditional likelihood of the available labels, ER also aims to minimize the entropy of the *predicted* label distribution on unlabeled data. By insisting on peaked, confident predictions, ER guides the decision boundary away from dense regions of input space. It is simple and compelling—no pre-clustering, no "auxiliary functions," tuning of only one meta-parameter and it is discriminative.

Jiao et al. (2006) apply this method to linear-chain CRFs and demonstrate encouraging accuracy improvements on a gene-name-tagging task. However, the method they present for calculating the gradient of the entropy takes substantially greater time than the traditional supervised-only gradient. Whereas supervised training requires only classic forward/backward, taking time $O(ns^2)$ (sequence length times the square of the number of labels), their training method takes $O(n^2s^3)$—a factor of $O(ns)$ more. This greatly reduces the practicality of using large amounts of unlabeled data, which is exactly the desired use-case.

This paper presents a new, more efficient entropy gradient derivation and dynamic program that has the same asymptotic time complexity as the gradient for traditional CRF training, $O(ns^2)$. In order to describe this calculation, the paper introduces the concept of *subsequence constrained entropy*—the entropy of a CRF for an observed data sequence when part of the label sequence is fixed. These methods will allow training on larger unannotated data set sizes than previously possible and support active

Proceedings of NAACL HLT 2007, Companion Volume, pages 109–112,
Rochester, NY, April 2007.

learning.

2 Semi-Supervised CRF Training

Lafferty et al. (2001) present linear-chain CRFs, a discriminative probabilistic model over observation sequences x and label sequences $Y = \langle Y_1..Y_n \rangle$, where $|x| = |Y| = n$, and each label Y_i has s different possible discrete values. For a linear-chain CRF of Markov order one:

$$p_\theta(Y|x) = \frac{1}{Z(x)} \exp\left(\sum_k \theta_k F_k(x, Y)\right),$$

where $F_k(x, Y) = \sum_i f_k(x, Y_i, Y_{i+1}, i)$, and the partition function $Z(x) = \sum_Y \exp(\sum_k \theta_k F_k(x, Y))$. Given training data $D = \langle d_1..d_n \rangle$, the model is trained by maximizing the log-likelihood of the data $L(\theta; D) = \sum_d \log p_\theta(Y^{(d)}|x^{(d)})$ by gradient methods (e.g. Limited Memory BFGS), where the gradient of the likelihood is:

$$\frac{\partial}{\partial \theta_k} L(\theta; D) = \sum_d F_k(x^{(d)}, Y^{(d)}) - \sum_d \sum_Y p_\theta(Y|x^{(d)}) F_k(x^{(d)}, Y).$$

The second term (the expected counts of the features given the model) can be computed in a tractable amount of time, since according to the Markov assumption, the feature expectations can be rewritten:

$$\sum_Y p_\theta(Y|x) F_k(x, Y) = \sum_i \sum_{Y_i, Y_{i+1}} p_\theta(Y_i, Y_{i+1}|x) f_k(x, Y_i, Y_{i+1}).$$

A dynamic program (the forward/backward algorithm) then computes in time $O(ns^2)$ all the needed probabilities $p_\theta(Y_i, Y_{i+1})$, where n is the sequence length, and s is the number of labels.

For semi-supervised training by *entropy regularization*, we change the objective function by adding the negative entropy of the unannotated data $U = \langle u_1..u_n \rangle$. (Here Gaussian prior is also shown.)

$$L(\theta; D, U) = \sum_n \log p_\theta(Y^{(d)}|x^{(d)}) - \sum_k \frac{\theta_k}{2\sigma^2} + \lambda \sum_u p_\theta(Y^{(u)}|x^{(u)}) \log p_\theta(Y^{(u)}|x^{(u)}).$$

This negative entropy term increases as the decision boundary is moved into sparsely-populated regions of input space.

3 An Efficient Form of the Entropy Gradient

In order to maximize the above objective function, the gradient for the entropy term must be computed. Jiao et al. (2006) perform this computation by:

$$\frac{\partial}{\partial \theta} - H(Y|x) = cov_{p_\theta(Y|x)}[F(x, Y)]\theta,$$

where

$$cov_{p_\theta(Y|x)}[F_j(x, Y), F_k(x, Y)] = E_{p_\theta(Y|x)}[F_j(x, Y), F_k(x, Y)] - E_{p_\theta(Y|x)}[F_j(x, Y)] E_{p_\theta(Y|x)}[F_k(x, Y)].$$

While the second term of the covariance is easy to compute, the first term requires calculation of quadratic feature expectations. The algorithm they propose to compute this term is $O(n^2 s^3)$ as it requires an extra nested loop in forward/backward.

However, the above form of the gradient is not the only possibility. We present here an alternative derivation of the gradient:

$$\begin{aligned}
&\frac{\partial}{\partial \theta_k} - H(Y|x) = \frac{\partial}{\partial \theta_k} \sum_Y p_\theta(Y|x) \log p_\theta(Y|x) \\
&= \sum_Y \left(\frac{\partial}{\partial \theta_k} p_\theta(Y|x)\right) \log p_\theta(Y|x) \\
&+ p_\theta(Y|x) \left(\frac{\partial}{\partial \theta_k} \log p_\theta(Y|x)\right) \\
&= \sum_Y p_\theta(Y|x) \log p_\theta(Y|x) \\
&\qquad \times \left(F_k(x, Y) - \sum_{Y'} p_\theta(Y'|x) F_k(x, Y')\right) \\
&+ \sum_Y p_\theta(Y|x) \left(F_k(x, Y) - \sum_{Y'} p_\theta(Y'|x) F_k(x, Y')\right).
\end{aligned}$$

Since $\sum_Y p_\theta(Y|x) \sum_{Y'} p_\theta(Y'|X) F_k(x, Y') = \sum_{Y'} p_\theta(Y'|X) F_k(x, Y')$, the second summand cancels, leaving:

$$\frac{\partial}{\partial \theta} - H(Y|x) = \sum_Y p_\theta(Y|x) \log p_\theta(Y|x) F_k(x, Y) - \left(\sum_Y p_\theta(Y|x) \log p_\theta(Y|x)\right) \left(\sum_{Y'} p_\theta(Y'|x) F_k(x, Y')\right).$$

Like the gradient obtained by Jiao et al. (2006), there are two terms, and the second is easily computable given the feature expectations obtained by

forward/backward and the entropy for the sequence. However, unlike the previous method, here the first term can be efficiently calculated as well. First, the term must be further factored into a form more amenable to analysis:

$$\begin{aligned}
&\sum_Y p_\theta(Y|x) \log p_\theta(Y|x) F_k(x, Y) \\
&= \sum_Y p_\theta(Y|x) \log p_\theta(Y|x) \sum_i f_k(x, Y_i, Y_{i+1}, i) \\
&= \sum_i \sum_{Y_i, Y_{i+1}} f_k(x, Y_i, Y_{i+1}, i) \\
&\qquad \sum_{Y_{-(i..i+1)}} p_\theta(Y|x) \log p_\theta(Y|x) \,.
\end{aligned}$$

Here, $Y_{-(i..i+1)} = \langle Y_{1..(i-1)} Y_{(i+2)..n} \rangle$. In order to efficiently calculate this term, it is sufficient to calculate $\sum_{Y_{-(i..i+1)}} p_\theta(Y|x) \log p_\theta(Y|x)$ for all pairs y_i, y_{i+1}. The next section presents a dynamic program which can perform these computations in $O(ns^2)$.

4 Subsequence Constrained Entropy

We define *subsequence constrained entropy* as

$$H^\sigma(Y_{-(a..b)}|y_{a..b}, x) = \sum_{Y_{-(a..b)}} p_\theta(Y|x) \log p_\theta(Y|x) \,.$$

The key to the efficient calculation for all subsets is to note that the entropy can be factored given a linear-chain CRF of Markov order 1, since Y_{i+2} is independent of Y_i given Y_{i+1}.

$$\begin{aligned}
&\sum_{Y_{-(a..b)}} p_\theta(Y_{-(a..b)}, y_{a..b}|x) \log p_\theta(Y_{-(a..b)}, y_{a..b}|x) \\
&\quad = \sum_{Y_{-(a..b)}} p_\theta(y_{a..b}|x) p_\theta(Y_{-(a..b)}|y_{a..b}, x) \times \\
&\qquad \big(\log p_\theta(y_{a..b}|x) + \log p_\theta(Y_{-(a..b)}|y_{a..b}, x)\big) \\
&\quad = p_\theta(y_{a..b}|x) \log p_\theta(y_{a..b}|x) \\
&\qquad + p_\theta(y_{a..b}|x) H^\sigma(Y_{-(a..b)}|y_{a..b}, x) \\
&\quad = p_\theta(y_{a..b}|x) \log p_\theta(y_{a..b}|x) \\
&\qquad + p_\theta(y_{a..b}|x) H^\alpha(Y_{1..(a-1)}|y_a, x) \\
&\qquad + p_\theta(y_{a..b}|x) H^\beta(Y_{(b+1)..n}|y_b, x) \,.
\end{aligned}$$

Given the $H^\alpha(\cdot)$ and $H^\beta(\cdot)$ lattices, any sequence entropy can be computed in constant time. Figure 1 illustrates an example in which the constrained sequence is of size two, but the method applies to arbitrary-length contiguous label sequences.

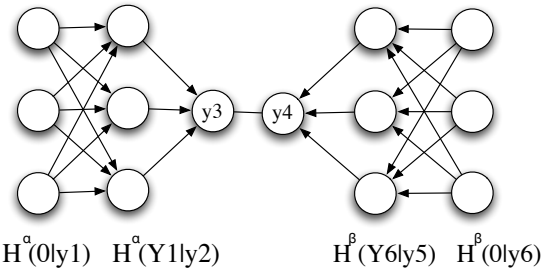


Figure 1: Partial lattice shown for computing the subsequence constrained entropy: $\sum_Y p(Y_{-(3..4)}, y_3, y_4) \log p(Y_{-(3..4)}, y_3, y_4)$. Once the complete H^α and H^β lattices are constructed (in the direction of the arrows), the entropy for each label sequence can be computed in linear time.

Computing the $H^\alpha(\cdot)$ and $H^\beta(\cdot)$ lattices is easily performed using the probabilities obtained by forward/backward. First recall the decomposition formulas for entropy:

$$\begin{aligned}
H(X, Y) &= H(X) + H(Y|X) \\
H(Y|X) &= \sum_x P(X = x) H(Y|X = x) \,.
\end{aligned}$$

Using this decomposition, we can define a dynamic program over the entropy lattices similar to forward/backward:

$$\begin{aligned}
&H^\alpha(Y_{1..i}|y_{i+1}, x) \\
&\quad = H(Y_i|y_{i+1}, x) + H(Y_{1..(i-1)}|Y_i, y_{i+1}, x) \\
&\quad = \sum_{y_i} p_\theta(y_i|y_{i+1}, x) \log p_\theta(y_i|y_{i+1}, x) \\
&\qquad + \sum_{y_i} p_\theta(y_i|y_{i+1}, x) H^\alpha(Y_{1..(i-1)}|y_i) \,.
\end{aligned}$$

The base case for the dynamic program is $H^\alpha(\emptyset|y_1) = p(y_1) \log p(y_1)$. The backward entropy is computed in a similar fashion. The conditional probabilities $p_\theta(y_i|y_{i-1}, x)$ in each of these dynamic programs are available by marginalizing over the per-transition marginal probabilities obtained from forward/backward.

The computational complexity of this calculation for one label sequence requires one run of forward/backward at $O(ns^2)$, and equivalent time to

calculate the lattices for H^α and H^β. To calculate the gradient requires one final iteration over all label pairs at each position, which is again time $O(ns^2)$, but no greater, as forward/backward and the entropy calculations need only to be done once. The complete asymptotic computational cost of calculating the entropy gradient is $O(ns^2)$, which is the same time as supervised training, and a factor of $O(ns)$ faster than the method proposed by Jiao et al. (2006).

Wall clock timing experiments show that this method takes approximately 1.5 times as long as traditional supervised training—less than the constant factors would suggest.[1] In practice, since the three extra dynamic programs do not require recalculation of the dot-product between parameters and input features (typically the most expensive part of inference), they are significantly faster than calculating the original forward/backward lattice.

5 Confidence Estimation

In addition to its merits for computing the entropy gradient, subsequence constrained entropy has other uses, including confidence estimation. Kim et al. (2006) propose using entropy as a confidence estimator in active learning in CRFs, where examples with the most uncertainty are selected for presentation to humans labelers. In practice, they approximate the entropy of the labels given the N-best labels. Not only could our method quickly and exactly compute the true entropy, but it could also be used to find the *sub*sequence that has the highest uncertainty, which could further reduce the additional human tagging effort.

6 Related Work

Hernando et al. (2005) present a dynamic program for calculating the entropy of a HMM, which has some loose similarities to the forward pass of the algorithm proposed in this paper. Notably, our algorithm allows for efficient calculation of entropy for any label subsequence.

Semi-supervised learning has been used in many models, predominantly for classification, as opposed to structured output models like CRFs. Zhu (2005) provides a comprehensive survey of popular semi-supervised learning techniques.

[1] Reporting experimental results with accuracy is unnecessary since we duplicate the training method of Jiao et al. (2006).

7 Conclusion

This paper presents two algorithmic advances. First, it introduces an efficient method for calculating subsequence constrained entropies in linear-chain CRFs, (useful for active learning). Second, it demonstrates how these subsequence constrained entropies can be used to efficiently calculate the gradient of the CRF entropy in time $O(ns^2)$—the same asymptotic time complexity as the forward/backward algorithm, and a $O(ns)$ improvement over previous algorithms—enabling the practical application of CRF *entropy regularization* to large unlabeled data sets.

Acknowledgements

This work was supported in part by DoD contract #HM1582-06-1-2013, in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0427594, and in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under contract number NBCHD030010. Any opinions, findings and conclusions or recommendations expressed in this material belong to the author(s) and do not necessarily reflect those of the sponsor.

References

Y. Grandvalet and Y. Bengio. 2004. Semi-supervised learning by entropy minimization. In *NIPS*.

D. Hernando, V. Crespi, and G. Cybenko. 2005. Efficient computation of the hidden markov model entropy for a given observation sequence. *IEEE Trans. on Information Theory*, 51:7:2681–2685.

F. Jiao, S. Wang, C.-H. Lee, R. Greiner, and D. Schuurmans. 2006. Semi-supervised conditional random fields for improved sequence segmentation and labeling. In *COLING/ACL*.

S. Kim, Y. Song, K. Kim, J.-W. Cha, and G. G. Lee. 2006. Mmr-based active machine learning for bio named entity recognition. In *HLT/NAACL*.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289.

X. Zhu. 2005. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.

Hybrid Document Indexing with Spectral Embedding

Irina Matveeva
Department of Computer Science
University of Chicago
Chicago, IL 60637
matveeva@cs.uchicago.edu

Gina-Anne Levow
Department of Computer Science
University of Chicago
Chicago, IL 60637
levow@cs.uchicago.edu

Abstract

Document representation has a large impact on the performance of document retrieval and clustering algorithms. We propose a hybrid document indexing scheme that combines the traditional bag-of-words representation with spectral embedding. This method accounts for the specifics of the document collection and also uses semantic similarity information based on a large scale statistical analysis. Clustering experiments showed improvements over the traditional *tf-idf* representation and over the spectral methods based solely on the document collection.

1 Introduction

Capturing semantic relations between words in a document representation is a difficult problem. Different approaches tried to overcome the term independence assumption of the bag-of-words representation (Salton and McGill, 1983) for example by using distributional term clusters (Slonim and Tishby, 2000) and expanding the document vectors with synonyms, see (Levow et al., 2005). Since content words can be combined into semantic classes there has been a considerable interest in low-dimensional term and document representations.

Latent Semantic Analysis (LSA) (Deerwester et al., 1990) is one of the best known dimensionality reduction algorithms. In the LSA space documents are indexed with latent semantic concepts. LSA showed large performance improvements over the traditional *tf-idf* representation on small document collections (Deerwester et al., 1990) but often does not perform well on large heterogeneous collections.

LSA maps all words to low dimensional vectors. However, the notion of semantic relatedness is defined differently for subsets of the vocabulary. In addition, the numerical information, abbreviations and the documents' style may be very good indicators of their topic. However, this information is no longer available after the dimensionality reduction.

We use a hybrid approach to document indexing to address these issues. We keep the notion of latent semantic concepts and also try to preserve the specifics of the document collection. We use a low-dimensional representation only for nouns and represent the rest of the document's content as *tf-idf* vectors.

The rest of the paper is organized as follows. Section 2 discusses our approach. Section 3 reports the experimental results. We conclude in section 4.

2 Hybrid Document Indexing

This section gives the general idea of our approach. We divide the vocabulary into two sets: nouns and the rest of the vocabulary. We use a method of spectral embedding, as described below and compute a low-dimensional representation for documents using only the nouns. We also compute a *tf-idf* representation for documents using the other set of words. Since we can treat each latent semantic concept in the low-dimensional representation as part of the vocabulary, we combine the two vector representations for each document by concatenating them.

Proceedings of NAACL HLT 2007, Companion Volume, pages 113–116,
Rochester, NY, April 2007.

2.1 Spectral Embedding

Spectral methods comprise a family of algorithms that use a matrix of pair-wise similarities S and perform its spectral analysis, such as the eigenvalue decomposition, to embed terms and documents in a low-dimensional vector space. $S = U\Sigma U^T$, where the columns of U are its eigenvectors and Σ is a diagonal matrix with the eigenvalues.

If we have a matrix of pair-wise word similarities S, its first k eigenvectors U_k will be used to represent the words in the latent semantic space. Semantically related words will have high association with the same latent concepts and their corresponding vectors will be similar. Moreover, the vector similarity between the word vectors will optimally preserve the original similarities (Cox and Cox, 2001).

We use two approaches to compute spectral embedding for nouns. Latent Semantic Analysis (LSA) (Deerwester et al., 1990) and Generalized Latent Semantic Analysis (GLSA) (Matveeva et al., 2005). For both we used the eigenvalue decomposition as the embedding step. The difference is in the similarities matrix which we are trying to preserve.

2.2 Distributional Term Similarity

LSA and GLSA begin with a matrix of pair-wise term similarities S, compute its eigenvectors U and use the first k of them to represent terms and documents, for details see (Deerwester et al., 1990; Matveeva et al., 2005). The main difference in our implementation of these algorithms is the matrix of pair-wise word similarities. Since our representation will try to preserve them it is important to have a matrix of similarities which is linguistically motivated.

LSA uses the matrix of pair-wise similarities which is based on document vectors. For two words w_i and w_j in the document collection containing n documents d_k, the similarity is computed as

$$S(w_i, w_j) =$$

$$\sum_{k=1:n} \mathrm{tf}(w_i, d_k)\mathrm{idf}(w_i) * \mathrm{tf}(w_j, d_k)\mathrm{idf}(w_j),$$

where $\mathrm{tf}(w_i, d_k)$ is the term frequency for w_i in d_k and $\mathrm{idf}(w_i)$ is the inverse document frequency weight for w_i. LSA is a special case of spectral embedding restricted to one type of term similarities and dimensionality reduction method.

GLSA (Matveeva et al., 2005) generalizes the idea of latent semantic space. It proposes to use different types of similarity matrix and spectral embedding methods to compute a latent space which is closer to true semantic similarities. One way to do so is to use a more appropriate similarities matrix S.

PMI We use point-wise mutual information (PMI) to compute the matrix S. PMI between random variables representing the words w_i and w_j is computed as

$$PMI(w_i, w_j) = \log \frac{P(W_i = 1, W_j = 1)}{P(W_i = 1)P(W_j = 1)}.$$

Thus, for GLSA, $S(w_i, w_j) = PMI(w_i, w_j)$.

Co-occurrence Proximity An advantage of PMI is the notion of proximity. The co-occurrence statistics for PMI are typically computed using a sliding window. Thus, PMI will be large only for words that co-occur within a small fixed context. Our experiments show that this is a better approximation to true semantic similarities.

2.3 Document Indexing

We have two sets of the vocabulary terms: a set of nouns, N, and the other words, T. We compute *tf-idf* document vectors indexed with the words in T:

$$\vec{d_i} = (\alpha_i(w_1), \alpha_i(w_2), ..., \alpha_i(w_{|T|})),$$

where $\alpha_i(w_t) = \mathrm{tf}(w_t, d_i) * \mathrm{idf}(w_t)$.

We also compute a k-dimensional representation with latent concepts c_i as a weighted linear combination of LSA or GLSA term vectors $\vec{w_t}$:

$$\vec{d_i} = (c_1, ..., c_k) = \sum_{t=1:|T|} \alpha_i(w_t) * \vec{w_t},$$

We concatenate these two representations to generate a hybrid indexing of documents:

$$\vec{d_i} = (\alpha_i(w_1), ..., \alpha_i(w_{|T|}), c_1, ...c_k)$$

3 Experiments

We performed document clustering experiments to validate our approach.

Subset m-n	#topics	min #d	max #d	av. #d
5-10	19	6	10	8.2
50-150	21	55	150	94.7
500-1000	2	544	844	694.0
1000-5000	3	1367	2083	1792.3

Table 1: TDT2 topic subsets containing between m and n documents: the number of topics per subset, the minimum, the maximum and the average number of documents per topic in each subset.

Indexing		
All words	Nouns	Hybrid
tf-idf, LSA GLSA, $GLSA\_local$	*tf-idf*$_N$ GLSA$_N$	*tf-idf*+GLSA$_N$

Table 2: Indexing schemes: with full vocabulary (All), only nouns (Nouns) and the combination.

Data We used the TDT2 collection[1] of news articles from six news agencies in 1998. We used only 10,329 documents that are assigned to one topic. TDT2 documents are distributed over topics very unevenly. We used subsets of the TDT2 topics that contain between m and n documents, see Table 1. We used the Lemur toolkit[2] with stemming and stop words list for the *tf-idf* indexing, Bikel's parser[3] to obtain the set of nouns and the PLAPACK package (Bientinesi et al., 2003) to compute the eigenvalue decomposition.

Global vs. Local Similarity To obtain the PMI values for GLSA we used the TDT2 collection, denoted as $GLSA_{local}$. Since co-occurrence statistics based on larger collections gives a better approximation to linguistic similarities, we also used 700,000 documents from the English GigaWord collection, denoted as GLSA and GLSA$_N$. We used a window of size 8.

Representations For each document we computed 7 representations, see Table 2. The vocabulary size we used with the *tf-idf* indexing was 114,127. For computational reasons we used the set of words that occurred in at least 20 documents with our spectral methods. We used 17,633 words for indexing with LSA and $GLSA_{local}$ and 17,572 words for GLSA. We also indexed documents using only the 15,325 nouns: *tf-idf*$_N$ and $GLSA_N$. The hybrid representation was computed using the *tf-idf* indexing without nouns and the $GLSA_N$ nouns vectors.

Evaluation We used the minimum squared residue co-clustering algorithm [4]. We report two evaluation measures: accuracy and the F1-score. The clustering algorithm assigns each document to a cluster. We map the cluster id's to topic labels using the Munkres assignment algorithm (Munkres, 1957) and compute the accuracy as the ratio of the correctly assigned labels.

The F1 score for cluster c_i labeled with topic t_i is computed using $F1 = \frac{2(p*r)}{(p+r)}$ where p is precision and r is recall. For clusters $C = (c_1, ..., c_n)$ and topics $T = (t_1, ..., t_n)$ we compute the total score:

$$F1(C,T) = \sum_{t \in T} \frac{N_t}{N} \max_{c \in C} F1(c,t).$$

N_t is the number of documents belonging to the topic t and N is the total number of documents. This measure accounts for the topic size and also corrects the topic assignments to clusters by using the max.

4 Results and Conclusion

Table 3 shows that the spectral methods outperform the *tf-idf* representations and have smaller variance. We report the performance for four subsets. The subset $5-10$ has a large number of topics, each with a similar number of documents. The subset $50-150$ has a large number of topics with a less even distribution of documents. $500-1000$ and $1000-5000$ have a couple of large topics. We ran the clustering over 30 random initializations. To eliminate the effect of the initial conditions on the performance we also used one document per cluster to seed the initial assignment for the $5-10$ subset.

All methods have the worst performance for the $5-10$ subset. The best performance is for the subset $500-1000$. LSA and $GLSA_{local}$ indexing are computed based on the TDT2 collection. $GLSA_{local}$ has better average performance which confirms that the co-occurrence proximity is important for distributional similarity. The GLSA indexing computed using a large corpus performs significantly worse than

[1] http://nist.gov/speech/tests/tdt/tdt98/

[2] http://www.lemurproject.org/

[3] http://www.cis.upenn.edu/ dbikel/software.html

[4] http://www.cs.utexas.edu/users/dml/Software/cocluster.html

		All words	LSA	GLSA$_{local}$	GLSA	onlyN	GLSA$_N$	Hybrid
5-10	acc	0.56(0.11)	0.69(0.07)	0.78(0.05)	0.60(0.05)	0.63(0.05)	0.76(0.05)	0.82(0.05)
	F1	0.60(0.09)	0.73(0.05)	0.81(0.04)	0.64(0.05)	0.67(0.05)	0.80(0.04)	0.85(0.04)
50-150	acc	0.75(0.05)	0.73(0.06)	0.80(0.05)	0.70(0.04)	0.68(0.04)	0.80(0.04)	0.87(0.04)
	F1	0.80(0.04)	0.78(0.05)	0.84(0.04)	0.75(0.04)	0.75(0.03)	0.84(0.04)	0.90(0.03)
500-1000	acc	0.95(0.03)	0.98(0.00)	0.99(0.00)	0.97(0.00)	0.97(0.00)	0.99(0.00)	1.00(0.00)
	F1	0.95(0.03)	0.98(0.00)	0.99(0.00)	0.97(0.00)	0.97(0.00)	0.99(0.00)	1.00(0.00)
1000-5000	acc	0.86(0.11)	0.88(0.04)	0.88(0.13)	0.92(0.08)	0.82(0.06)	0.92(0.00)	0.96(0.07)
	F1	0.88(0.07)	0.88(0.03)	0.90(0.09)	0.93(0.06)	0.82(0.04)	0.92(0.00)	0.97(0.05)
5-10$_s$	acc	0.932	0.919	0.986	0.932	0.980	0.980	0.992
	F1	0.933	0.927	0.986	0.932	0.979	0.979	0.992

Table 3: Clustering accuracy (first row) and F1 score (second row) for each indexing scheme. The measures are averaged over 30 random initiations of the clustering algorithm, the standard deviation is shown in brackets. For the last experiment, 5-10$_s$, we used one document per cluster as the initial assignment.

$GLSA_{local}$ on the heterogeneous $5-10$ and $50-150$ subsets and performs similarly for the other two. It supports our intuition that the document's style and word distribution within the collection are important and may get lost, especially if we use a document collection with a different word distribution to estimate the similarities matrix S.

The *tf-idf* indexing with nouns only, $onlyN$, has good performance compared to the all-words indexing. The semantic similarity between nouns seems to be collection independent. The $GLSA_N$ indexing is significantly better than $onlyN$ and *tf-idf* in most cases and performs similar to $GLSA_{local}$. By using $GLSA_N$ we computed the embedding for more nouns that we could keep in the $GLSA_{local}$ and $GLSA$ representations. Nouns convey important topic membership information and it is advantageous to use as many of them as possible.

We observed the same performance relation when we used labels to make the initial cluster assignment, see $5-10_s$ in Table 3. *tf-idf*, GLSA and LSA performed similarly, $GLSA_{local}$ and $GLSA_N$ performed better with the hybrid scheme being the best.

The hybrid indexing significantly outperforms *tf-idf*, LSA and GLSA on three subsets. This shows the benefits of using the spectral embedding to discover the semantic relations between nouns and keeping the rest of the document content as *tf-idf* representation to preserve other indicators of its topic membership. By combining two representations the hybrid indexing scheme defines a more complex notion of similarity between documents. For nouns it uses the semantic proximity in the space of latent semantic classes and for other words it uses term-matching.

References

Paolo Bientinesi, Inderjit S. Dhilon, and Robert A. van de Geijn. 2003. A parallel eigensolver for dense symmetric matrices based on multiple relatively robust representations. *UT CS Technical Report TR-03-26*.

Trevor F. Cox and Micheal A. Cox. 2001. *Multidimensional Scaling*. CRC/Chapman and Hall.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407.

Gina-Anne Levow, Douglas W. Oard, and Philip Resnik. 2005. Dictionary-based techniques for cross-language information retrieval. *Information Processing and Management: Special Issue on Cross-language Information Retrieval*.

Irina Matveeva, Gina-Anne Levow, Ayman Farahat, and Christian Royer. 2005. Generalized latent semantic analysis for term representation. In *Proc. of RANLP*.

J. Munkres. 1957. Algorithms for the assignment and transportation problems. *SIAM*, 5(1):32–38.

Gerard Salton and Michael J. McGill. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill.

Noam Slonim and Naftali Tishby. 2000. Document clustering using word clusters via the information bottleneck method. In *Research and Development in Information Retrieval*, pages 208–215.

On using Articulatory Features for Discriminative Speaker Adaptation

Florian Metze
Deutsche Telekom Laboratories
Berlin; Germany
florian.metze@telekom.de

Abstract

This paper presents a way to perform speaker adaptation for automatic speech recognition using the stream weights in a multi-stream setup, which included acoustic models for "Articulatory Features" such as ROUNDED or VOICED. We present supervised speaker adaptation experiments on a spontaneous speech task and compare the above stream-based approach to conventional approaches, in which the models, and not stream combination weights, are being adapted. In the approach we present, stream weights model the importance of features such as VOICED for word discrimination, which offers a descriptive interpretation of the adaptation parameters.

1 Introduction

Almost all approaches to automatic speech recognition (ASR) using Hidden Markov Models (HMMs) to model the time dependency of speech are also based on phones, or context-dependent sub-phonetic units derived from them, as the atomic unit of speech modeling. In phonetics, a phone is a shorthand notation for a certain configuration of underlying articulatory features (AFs) (Chomsky and Halle, 1968): /p/ is for example defined as the unvoiced, bi-labial plosive, from which /b/ can be distinguished by its VOICED attribute. In this sense, instead of describing speech as a single, sequential stream of symbols representing sounds, we can also look at speech as the result of a process involving several parallel streams of information, each of which describes some linguistic or articulatory property as being either absent or present.

A multi-stream architecture is a relatively simple approach to combining several information sources in ASR, because it leaves the basic structure of the Hidden Markov Model and its computational complexity intact. Examples combining different observations are audio-visual speech recognition (Potamianos and Graf, 1998) and sub-band based speech processing (Janin et al., 1999). The same idea can also be used to combine different classifiers on the same observation. In a multi-stream HMM setup, *log-linear interpolation* (Beyerlein, 2000) can be derived as a framework to integrating several independent acoustic models given as Gaussian Mixture Models (GMMs) into the speech recognition process: given a "weight" vector $\Lambda = \{\lambda_0, \lambda_1, \cdots, \lambda_M\}$, a word sequence W, and an acoustic observation $\mathbf{o}$, the posterior probability $p(W|\mathbf{o})$ one wants to optimize is written as:

$$p(W|\mathbf{o}) = C \exp\left\{\sum_{i=0}^{M} \lambda_i \log p_i(W|\mathbf{o})\right\}$$

C is a normalization constant, which can be neglected in practice, as long as normalization $\Sigma_i \lambda_i = const$ is observed. It is now possible to set $p(W|\mathbf{o}) \propto p(\mathbf{o}|W)$ (Beyerlein, 2000) and write a speech recognizer's acoustic model $p(\mathbf{o}|W)$ in this form, which in logarithmic representation reduces to a simple weighted sum of so-called "scores" for each individual stream. The λ_i represent the "im-

Proceedings of NAACL HLT 2007, Companion Volume, pages 117–120,
Rochester, NY, April 2007.

portance" of the contribution of each individual information source.

Extending Kirchhoff's (Kirchhoff, 1999) approach, the log-likelihood score combination method to AF-based ASR can be used to combine information from M different articulatory features while at the same time retaining the "standard" acoustic models as stream 0. As an example using $M = 2$, the acoustic score for /z/ would be computed as a weighted sum of the scores for a (context-dependent sub-)phonetic model z, the score for FRICATIVE and the score for VOICED, while the score for /s/ would be computed as a weighted sum of the scores for a (context-dependent sub-) phonetic model s, the score for FRICATIVE and the score for NON_VOICED. The free parameters λ_i can be global (G), or they can be made state-dependent (SD) during the optimization process, thus changing the importance of a feature given a specific phonetic context, as long as overall normalization is observed. (Metze, 2005) discusses this stream setup in more detail.

2 Experiments

To investigate the performance of the proposed AF-based model, we built acoustic models for 68 articulatory features on 32h of English Spontaneous Scheduling Task ESST data from the Verbmobil project (Wahlster, 2000), and integrated them with matching phone-based acoustic models.

For training robust baseline phone models, 32h from the ESST corpus were merged with 66h Broadcast News '96 data, for which manually annotated speaker labels are available. The system is trained using 6 iterations of ML training and uses 4000 context dependent (CD) acoustic models (HMM states), 32 Gaussians per model with diagonal covariance matrices and a global semi-tied covariance matrix (STC) in a 40-dimensional MFCC-based feature space after LDA. The characteristics of the training and test sets used in the following experiments are summarized in Table 1.

The ESST test vocabulary contains 9400 words including pronunciation variants (7100 base forms) while the language model perplexity is 43.5 with an out of vocabulary (OOV) rate of 1%. The language model is a tri-gram model trained on ESST data containing manually annotated semantic classes for most proper names (persons, locations, numbers). Generally, systems run in less than 4 times real-time on Pentium 4-class machines. The baseline Word Error Rate is reported as adaptation "None" in Table 2; the system parameters were optimized on the ds2 data set. As the stream weight estimation process can introduce a scaling factor for the acoustic model, we verified that the baseline system can not be improved by widening the beam or by readjusting the weight of the language model vs. the acoustic model. The baseline system can also not be improved significantly by varying the number of parameters, either by increasing the number of Gaussians per codebook or by increasing the number of codebooks.

Data Set	Train	Test		
		1825	ds2	xv2
Duration	98h	2h25	1h26	0h59
Utterances	39100	1825	1150	675
Recordings	8681	58	32	26
Speakers	423	16	9	7

Table 1: Data sets used in this work: The ESST test set 1825 is the union of the development set ds2 and the evaluation set xv2.

2.1 MMI Training of Stream Weights

To arrive at an optimal set of stream weights, we used the iterative update rules presented in (Metze, 2005) to generate stream weights λ_i using the Maximum Mutual Information (MMI) criterion (Bahl et al., 1986).

Results after one iteration of stream weight estimation on the 1825 and ds2 data sets using step size $\epsilon = 4 \cdot 10^{-8}$, initial stream weight $\lambda^0_{i\neq 0} = 3 \cdot 10^{-3}$, and lattice density $d = 10$ are shown in Table 2 in rows "AF (G) on 1825" and "AF (G) on ds2": As there are only 68 stream weights to estimate, adaptation works only slightly better when adapting and testing on the same corpus ("cheating experiment": 22.6% vs. 22.8% word error rate (WER) on ds2). There is no loss in WER (24.9%) on xv2 when adapting the weights on ds2 instead of 1825, which has no overlap with xv2, so generalization on unseen test data is good for global

stream weights, i.e. weights which do not depend on state or context.

2.2 Speaker-specific Stream Weights

The ESST test `1825` set is suitable to test speaker-specific properties of articulatory features, because it contains 16 speakers in 58 different recordings. As `1825` provides between 2 and 8 dialogs per speaker, it is possible to adapt the system to individual speakers in a "round-robin" or "leave-one-out" experiment, i.e. to decode every test dialog with weights adapted on all remaining dialogs from that speaker in the `1825` test set. Using speaker-specific, but global (G), weights computed with the above settings, the resulting WER is 21.5% (row "AF (G) on speaker" in Table 2).

Training parameters were chosen to display improvements after the first iteration of training without convergence in further iterations. Consequently, training a second iteration of global (i.e. context independent) weights does not improve the performance of the speaker adapted system. In our experiments we reached best results when computing state-dependent (SD) feature weights on top of global weights using the experimentally determined smaller learning rate of $\epsilon_{SD} = 0.2 \cdot \epsilon$. In this case, speaker and state dependent AF stream weights further reduce the word error rate to 19.8% (see bottom row of Table 2).

2.3 ML Model Adaptation

When training speaker-dependent articulatory feature weights in Section 2.2, we were effectively performing supervised speaker adaptation (on separate adaptation data) with articulatory feature weights. To compare the performance of AFs to other approaches to speaker adaptation, we adapted the baseline acoustic models to the test data using supervised maximum likelihood linear regression (MLLR) (Leggetter and Woodland, 1994) and constrained MLLR, which is also known as "feature-space adaptation" (FSA) (Gales, 1997).

The ESST data has very little channel variation so that the performance of models that were trained on both ESST and BN data can be improved slightly on ESST test dialogs by using FSA, while MLLR already leads to over-specialization (Table 2, rows "FSA/ MLLR on `ds2`). The results in Table 2 show that AF adaptation performs as well as FSA in the case of supervised adaptation on the `ds2` data and better by about 1.3% absolute in the speaker adaptation case, despite using significantly less parameters (69 for the AF case vs. 40*40=1.6k for the FSA case). While supervised FSA is equivalent to AF adaptation when adapting and decoding on the `ds2` data in a "cheating experiment" for diagnostic purposes (22.5% vs 22.6%, rows "FSA/ AF (G) on `ds2`" of Table 2), supervised FSA only reaches a WER of 22.8% on `1825` when decoding every ESST dialog with acoustic models adapted to the other dialogs available for this speaker (row "FSA on speaker"). AF-based adaptation reaches 21.5% for the global (G) case and 19.8% for the state dependent (SD) case (last two rows). The AF (SD) case has 68*4000=276k free parameters, but decision-tree based tying using a minimum count reduces these to 4.3k per speaker. Per-speaker MLLR uses 4.7k parameters in the transformation matrices on average per speaker, but performs worse than AF-based adaptation by about 1% absolute.

Adaptation type and corpus	Test corpus `1825`	`ds2`	`xv2`
None	25.0%	24.1%	26.1%
FSA on `ds2`		22.5%	25.4%
FSA on speaker	22.8%	21.6%	24.3%
MLLR on `ds2`		16.3%	26.4%
MLLR on speaker	20.9%	19.8%	22.4%
MMI-MAP on `ds2`		14.4%	26.2%
MMI-MAP on speaker	20.5%	19.5%	21.7%
AF (G) on `1825`	23.7%	22.8%	24.9%
AF (G) on `ds2`		22.6%	24.9%
AF (SD) on `ds2`		22.5%	26.5%
AF (G) on speaker	21.5%	20.1%	23.6%
AF (SD) on speaker	19.8%	18.6%	21.7%

Table 2: Word error rates on the ESST test sets using different kinds of adaptation. See Table 1 for a description of data sets.

2.4 MMI Model Adaptation

In a non-stream setup, discriminative speaker adaptation approaches have been published using conditional maximum likelihood linear regression (CMLLR) (Gunawardana and Byrne, 2001) and MMI-

MAP (Povey et al., 2003). In supervised adaptation experiments on the Switchboard corpus, which are similar to the experiments presented in the previous section, CMLLR reduced word error rate over the baseline, but failed to outperform conventional MLLR adaptation (Gunawardana and Byrne, 2001), which was already tested in Section 2.3. We therefore compared AF-based speaker adaptation to MMI-MAP as described in (Povey et al., 2003).

The results are given in Table 2: using a comparable number of parameters for adaptation as in the previous section, AF-based adaptation performs slightly better than MMI-MAP (19.8% WER vs. 20.5%; rows "MMI-MAP/ AF (SD) on speaker"). When testing on the adaptation data `ds2` as a diagnostic experiment, MMI-MAP as well as MLLR outperform AF based adaptation, but the gains do not carry over to the validation set `xv2`, which we attribute to over-specialization of the acoustic models (rows "MLLR/ MMI-MAP/ AF (SD) on `ds2`).

3 Summary and Conclusion

This paper presented a comparison between two approaches to discriminative speaker adaptation: speaker adaptation using articulatory features (AFs) in the multi-stream setup presented in (Metze, 2005) slightly outperformed model-based discriminative approaches to speaker adaptation (Gunawardana and Byrne, 2001; Povey et al., 2003), however at the cost of having to evaluate additional codebooks in the articulatory feature streams during decoding. In our experiments, we used 68 AFs, which requires the evaluation of 68 models for "feature present" and 68 models for "feature absent" for each frame during decoding, plus the computation necessary for stream combination. In this setup however, the adaptation parameters, which are given by the stream combination weights, have an intuitive meaning, as they model the importance of phonological features such as VOICED or ROUNDED for word discrimination for this particular speaker and phonetic context. Context-dependent stream weights can also model feature asynchrony to some extent, so that this approach not only improves automatic speech recognition, but might also be an interesting starting point for future work in speaker clustering, speaker identification, or other applications in speech analysis.

References

Lalit R. Bahl, Peter F. Brown, Peter V. de Souza, and Robert L. Mercer. 1986. Maximum mutual information estimation of Hidden Markov Model parameters for speech recognition. In *Proc. ICASSP*, volume 1, pages 49–52, Tokyo; Japan, May. IEEE.

Peter Beyerlein. 2000. *Diskriminative Modellkombination in Spracherkennungssystemen mit großem Wortschatz*. Ph.D. thesis, Rheinisch-Westfälisch-Technische Hochschule Aachen (RWTH), October. In German.

Noam Chomsky and Morris Halle. 1968. *The Sound Pattern of English*. Harper and Row, New York; USA.

Mark J. F. Gales. 1997. Maximum likelihood linear transformations for HMM-based speech recognition. Technical report, Cambridge University, Cambridge; UK, May. CUED/F-INFENG/TR 291.

Asela Gunawardana and William Byrne. 2001. Discriminative speaker adaptation with conditional maximum likelihood linear regression. In *Proc. Eurospeech 2001 - Scandinavia*, Aalborg; Denmark, September. ISCA.

Adam Janin, Dan Ellis, and Nelson Morgan. 1999. Multi-stream speech recognition: Ready for prime time. In *Proc. EuroSpeech 1999*, Budapest; Hungary, September. ISCA.

Katrin Kirchhoff. 1999. *Robust Speech Recognition Using Articulatory Information*. Ph.D. thesis, Technische Fakultät der Universität Bielefeld, Bielefeld; Germany, June.

Chris J. Leggetter and Phil C. Woodland. 1994. Speaker adaptation of HMMs using linear regression. Technical report, Cambridge University, England.

Florian Metze. 2005. *Articulatory Features for Conversational Speech Recognition*. Ph.D. thesis, Fakultät für Informatik der Universität Karlsruhe (TH), Karlsruhe; Germany, December.

Gerasimos Potamianos and Hans-Peter Graf. 1998. Discriminative training of HMM stream exponents for audio-visual speech recognition. In *Proc. ICASSP 1998*, Seattle, WA; USA. IEEE.

Dan Povey, Mark J.F. Gales, Do Y. Kim, and Phil C. Woodland. 2003. MMI-MAP and MPE-MAP for acoustic model adaptation. In *Proc. Eurospeech 2003*, Geneva; Switzerland, September. ISCA.

Wolfgang Wahlster, editor. 2000. *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, Heidelberg.

RH: A Retro Hybrid Parser

Paula S. Newman
newmanp@acm.org

Abstract

Contemporary parser research is, to a large extent, focused on statistical parsers and deep-unification-based parsers. This paper describes an alternative, hybrid architecture in which an ATN-like parser, augmented by many preference tests, builds on the results of a fast chunker. The combination is as efficient as most stochastic parsers, and accuracy is close and continues to improve. These results raise questions about the practicality of deep unification for symbolic parsing.

1 Introduction

The original goals of the RH parser were to obtain accurate parses where (a) application speed was needed, and (b) large amounts of annotated material for a subject idiom were not available. Additional goals that evolved were (c) that parses for particular documents could be brought to an almost arbitrary level of correctness for research purposes, by grammar correction, and (d) that information collected during parsing could be modified for an application with a modest amount of effort. Goal (a) ruled out the use of unification-based symbolic parsers, because deep unification is a relatively slow operation, no matter what amount of computational sophistication is employed. Until very recently, goal (b) ruled out stochastic parsers, but new results (McClosky et al. 2006) suggest this may no longer be the case. However, the "additional" goals still favor symbolic parsing.

To meet these goals, the RH parser combines a very efficient shallow parser with an overlay parser that is "retro", in that the grammar is related to Augmented Transition Networks (Woods, 1970), operating on the shallow-parser output. A major "augmentation" is a preference-scoring component.

Section 2 below reviews the shallow parser used, and Section 3 describes the overlay parser. Some current results are presented in section 4. Section 5 examines some closely-related work, and Section 6 discusses some implications.

2 The XIP Parser for English

XIP is a robust parser developed by Xerox Research Center Europe. It is actually a full parser that produces a tree of chunks, plus identification of (sometimes alternative) typed dependencies among the chunk heads (Ait-Mokhtar et al. 2002, Gala 2004). But because the XIP dependency analyzer for English was incomplete when RH work began, and because classic parse trees are more convenient for discourse-related applications, we focused on the chunk output.

XIP is astonishingly fast, contributing very little to RH parse time. It consists of the XIP engine, plus language-specific grammars, each consisting of: (a) a finite state lexicon producing alternative tags and morphological analyses for each token, together with subcategorization, control and (some) semantic class features, (b) a part of speech tagger, and (c) conveniently expressed, layered rule sets that perform the following functions:

- **Lexicon extension**, which adds words and adds or overrides feature information,
- **Lexical disambiguation** (including use of the tagger to provide default assignments)
- **Multi-word identification** for named entities, dates, short constructions, etc.
- **Chunking,** obtaining basic chunks such as basic adjective, adverbial, noun and prepositional phrases.
- **Dependency Analysis** (not used in RH)

All rule sets have been extended within RH development except for the dependency rule sets..

3 Overlay Parser

The overlay parser builds on chunker output to produce a single tree (figure 1) providing syntactic categories and functions, heads, and head features. The output tree requires further processing to obtain long distance dependency information, and make some unambiguous coordination adjustments

Proceedings of NAACL HLT 2007, Companion Volume, pages 121–124,
Rochester, NY, April 2007.

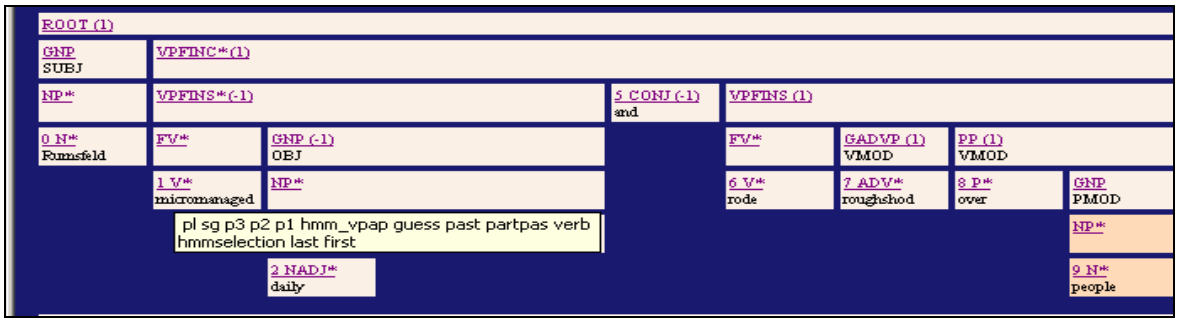


Figure 1. Output Parse Tree. * indicates head. Mouseover shows head features

Some of this has already been done in a post-parse phase. The feasibility of such post-parse deepening (for a statistical parser) is demonstrated by Cahill et al (2004).

The major parser components are a control, the ATN-like grammar networks, and collections of tests. The control is invoked recursively to build non-chunk constituents by following grammar network paths and creating output networks.

Figure 2 shows the arcs of an excerpt from a grammar network used to build a noun phrase. The Test labels on the arcs resemble specialized categories. The MetaOps (limited in the illustration to Prolog-like cuts) expedite processing by permitting or barring exploration of further ordered arcs originating at the same state.

An output network, illustrated in figure 3, mirrors the full paths traversed in a grammar network by one invocation of the control. The arcs refer either to chunks or to final states of other output networks. Output networks do not contain cycles or converging arcs, so states represent unique paths. They carry head and other path information, and a preference score. The final parser output is a single tree, derived from a highest scoring path of a topmost output network. Ties are broken by low attach considerations.

From	To	Test	Syn fun	Final?	Meta Op
S1	S1	PREADV	PRE	No	cut
S1	S2	PRON	HEAD	Yes	cut
S1	S3	PROPER	HEAD	Yes	cut
S1	S4 S7	BASENP	HEAD	Yes	cut
//After pronoun					
S2	-	REFL	REFL	Yes	cut
S2	-	PEOPLE	APPS	Yes	cut

Figure 2. Some arcs of grammar network for GNP

From	To	Cat	Synfun	Ref
OSa	OSb	NP	HEAD	NPChunk (*The park*)
OSb	OSc	PP	NMOD	Final state of PP net for (*in Paris*)
States		**Score**	**Final?**	
Osa		0	No	
Osb		0	Yes	
OSc		1	Yes	

Figure 3. Output network for "*The park in Paris*"

Each invocation of the control is given a grammar network entry state and a desired constituent category. After initializing a new output network, the arcs from the given entry state are followed. Processing an arc may begin with an optional pretest. If that succeeds, or there is no pretest, a constructive test follows. The tests are indexed by grammar network test labels, and are expressed as blocks of procedural code, for initial flexibility in determining the necessary checks.

Pretests include fast feasibility checks, and contexted checks of consistency of the potential new constituent with the current output network path. Constructive tests can make additional feasibility checks. If these checks succeed, either a chunk is returned, or the control is reentered to try to build a subordinate output network. Results are cached, to avoid repeated testing.

After a chunk or subordinate network *ON'* is returned from a constructive test, one new arc *Ai* is added to the current output network *ON* to represent each full path through *ON'*. All added arcs have the same origin state in *ON*, but unique successor states and associated preference scores. The preference score is the sum of the score at the common origin state, plus the score of the represented path in *ON'*, plus a contexted score for the alternative within *ON*. The latter is one of <-1, 0, +1>, and expresses the consistency of *Ai* with the current path with respect to dependency, coordination and apposition. Structural and punctuation

aspects are also considered. Preference tests are indexed by syntactic category or syntactic function, and are organized for speed. Most tests are independent of *Ai* length, and can be applied once and the results assumed for all *Ai*.

Before a completed output network is returned, paths ending at those lower scoring final states which cannot ultimately be optimal are pruned. Such pruning is critical to efficiency.

4 Indicative Current Results

To provide a snapshot of current RH parser performance, we compare its current speed and accuracy directly to those of a widely used statistical parser, Collins model 3 (Collins, 1999), and indirectly to two other parsers. Wall Street Journal section 23 of the Penn Treebank (Marcus et al. 1994) was used in all experiments.

"Training" of the RH parser on the Wall Street Journal area (beyond general RH development) occupied about 8 weeks, and involved testing and (non-exhaustively) correcting the parser using two WSJ texts: (a) section 00, and (b) 700 sentences of section 23 used as a dependency bank by King et al. (2003). The latter were used early in RH development, and so were included in the training set.

4.1 Comparative Speed

Table 1 compares RH parser speed with Collins model 3, using the same CPU, showing the elapsed times for the entire 2416-line section 23.

The results are then extrapolated to two other parsers, based on published comparisons with Collins. The extrapolation to XLE, a mature unification-based parser that uses a disambiguating statistical post-processor, is drawn from Kaplan et al. (2004). Results are given for both the full grammar and a reduced version that omits less likely rules. The second comparison is with the fast stochastic parser by Sagae and Lavie (2005).

Summarizing these results, RH is much faster than Collins model 3 and the reduced version of XLE, but a bit slower than Sagae-Lavie.

The table also compares coverage, as percentages of non-parsed sentences. For RH this was 10% for the test set discussed below, which did not contain any training sentences, and was 10.4% for the full section 23. This is reasonable for a symbolic parser with limited training on an idiom, and better than the 21% reported for XLE English.

	Time	No full parse
Sagae/ Lavie	***~ 4 min***	*1.1%*
RH parser	5 min	10%
Collins m3	16 min	**.6%**
XLE full	*~80 minutes*	~21%
XLE reduced	*~24 minutes*	unknown

Table 1: Speeds and *Extrapolated speeds*

	Fully accurate	F-score	Avg cross brackets
Sagae/Lavie	unknwn	86%	unknwn
Collins Lbl	33.6%	88.2%	1.05
CollinsNoLbl	35.4%	**89.4 %**	1.05
RH NoLbl	**46%**	86 %	**.59**

Table 2. Accuracy Comparison

4.2 Comparative Acccuracy

Table 2 primarily compares the accuracy of the Collins model 3 and RH parsers. The entries show the proportion of fully accurate parses, the f-score average of bracket precision and recall, and average crossing brackets, as obtained by EVALB (Sekine and Collins, 1997). The RH f-score is currently somewhat lower, but the proportion of fully correct parses is significantly higher.

This data may be biased toward RH, because, of necessity, the test set used is smaller, and a different bracketing method is used. For Collins model 3, the entries show both labeled and unlabeled results for all of WSJ section 23. The Collins results were generated from the bracketed output and Penn Treebank gold standard files provided in a recent Collins download.

But because RH does not generate treebank style tags, the RH entries reflect a test only on a random sample of 100 sentences from the 1716 sentences of section 23 not used as "training" data, using a different, available, gold standard creation and bracketing method. In that method (Newman, 2005), parser results are produced in a "TextTree" form, initially developed for fast visual review of parser output, and then edited to obtain gold standard trees. Both sets of trees are then bracketed by a script to obtain, e.g.,

{An automatic transformation
{of parse trees}
{to text trees}}
{can expedite
{parser output reviews}}

For non-parsed sentences in the parser outputs, brackets are applied to the chunks. EVALB is then used to compare the two sets of bracketed results.

Accuracy for XLE is not given, because the results reported by Kaplan et al. (2004) compare labeled functional dependencies drawn from LFG f-structures with equivalents derived automatically from Collins outputs. (All f-scores are <= 80%).

5 Related Work

Several efforts combine a chunker with a dependency analyzer operating on the chunks, including XIP itself. The XIP dependency analyzer is very fast, but we do not have current coverage or accuracy data for XIP English.

Other related hybrids do not build on chunks, but, rather, adjust full parsers to require or prefer results consistent with chunk boundaries. Daum et al. (2003) use chunks to constrain a WCDG grammar for German, reducing parse times by about 2/3 (but the same results are obtained using a tagger alone). They estimate that an ideal chunker would reduce times by about 75%. No absolute numbers are given. Also, Frank et al. (2003) use a German topological field identifier to constrain an HPSG parser. They show speedups of about 2.2 relative to a tagged baseline, on a corpus whose average sentence length is about 9 words.

6 Discussion

We have shown that the RH hybrid can compete with stochastic parsers in efficiency and, with only limited "training" on an idiom, can approach them in accuracy. Also, the test organization prevents speed from degrading as the parser is improved.

The method is significant in itself, but also leads to questions about the advantages of deep-unification-based parsers for practical NLP. These parsers are relatively slow, and their large numbers of results require disambiguation, e.g., by corpus-trained back-ends. They do provide more information than RH, but there is much evidence that the additional information can be obtained by rapid analysis of a single best parse. Also, it has never been shown that their elegant notations actually facilitate grammar development and maintenance. Finally, while unification grammars are reversible for use in generation, good generation methods remain an open research problem.

References

Salah Aït-Mokhtar, Jean-Pierre Chanod, and Claude Roux. 2002. Robustness beyond shallowness: incremental deep parsing, *Natural Language Engineering* 8:121-144, Cambridge University Press.

Aoife Cahill, Michael Burke, Ruth O'Donovan, Josef van Genabith, and Andy Way. 2004. Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-Based LFG Approximations, In *Proc ACL'04.* Barcelona

Michael Collins. 1999. Head-Driven Statistical Models for Natural Language Parsing. Ph.D. thesis, University of Pennsylvania.

Michael A. Daum, Kilian A. Foth, and Wolfgang Menzel. 2003. Constraint-based integration of deep and shallow parsing techniques. In *Proc EACL'03*, Budapest

Anette Frank, Markus Becker, Berthold Crysmann, Bernd Kiefer and Ulrich Schaefer. 2003. Integrated Shallow and Deep Parsing: TopP Meets HPSG. In *Proc ACL'2003,* Sapporo

Nuria Gala. 2004. Using a robust parser grammar to automatically generate UNL graphs. *In Proc Workshop on Robust Methods for Natural Language Data at COLING'04,* Geneva

Ronald M. Kaplan, Stephan Riezler, Tracy H. King, John T. Maxwell, Alex Vasserman. 2004. Speed and accuracy in shallow and deep stochastic parsing. In *Proc HLT/NAACL'04*, Boston, MA.

Tracy H. King, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald M. Kaplan. 2003. The PARC 700 dependency bank. In *Proc Workshop on Linguistically Interpreted Corpora, (LINC'03)*, Budapest

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Reranking and Self-Training for Parser Adaptation. In *Proc ACL'06.* Sydney

Paula Newman. 2005. TextTree Construction for Parser and Grammar Development. In *Proc. Workshop on Software at ACL'05* Ann Arbor, MI. Available at http://www.cs.columbia.edu/nlp/acl05soft/

Satoshi Sekine and Michael Collins. 1997. *EvalB.* Available at http://nlp.cs.nyu.edu/evalb

Kenji Sagae and Alon Lavie. 2005. A classifier-based parser with linear run-time complexity. In *Proc. 9th Int'l Workshop on Parsing Technologies*. Vancouver

William Woods. 1970. Transition network grammars for natural language analysis. *Communications of the ACM* 13(10), 591-606

Subtree Mining for Relation Extraction from Wikipedia

Dat P.T. Nguyen
University of Tokyo
7-3-1 Hongo, Bunkyo-ku
Tokyo 113-8656, Japan
nptdat@mi.ci.i.u-tokyo.ac.jp

Yutaka Matsuo
National Institute of Advanced
Industrial Science and Technology
Sotokanda 1-18-13
Tokyo 101-0021, Japan
y.matsuo@aist.go.jp

Mitsuru Ishizuka
University of Tokyo
7-3-1 Hongo, Bunkyo-ku
Tokyo 113-8656, Japan
ishizuka@i.u-tokyo.ac.jp

Abstract

In this study, we address the problem of extracting relations between entities from Wikipedia's English articles. Our proposed method first anchors the appearance of entities in Wikipedia's articles using neither Named Entity Recognizer (NER) nor coreference resolution tool. It then classifies the relationships between entity pairs using SVM with features extracted from the web structure and subtrees mined from the syntactic structure of text. We evaluate our method on manually annotated data from actual Wikipedia articles.

1 Introduction

Wikipedia (www.wikipedia.org) has emerged as the world's largest online encyclopedia. Because the encyclopedia is managed by the Wikipedia Foundation, and because numerous collaborators in the world continuously develop and edit its articles, its contents are believed to be quite reliable despite its openness.

This study is intended to deal with the problem of extracting binary relations between entity pairs from Wikipedia's English version. A binary relation is defined as a triple (e_p, *rel*, e_s) in which e_p and e_s are entities and *rel* indicates a directed relationship of e_p and e_s. Current experiment limits entities and relations to a reasonable size in that an entity is classifiable as *person*, *organization*, *location*, *artifact*, *year*, *month* or *date*; and a relation can be *founder*, *chairman*, *CEO*, *COO*, *president*, *director*, *vice chairman*, *spouse*, *birth date*, *birth place*, *foundation*, *product* and *location*.

To our knowledge, only one recent work has attempted relation extraction on Wikipedia: (Culotta et al., 2006) presents a probabilistic model to integrate extraction and mining tasks performed on biographical text of Wikipedia. Some other works (Brin, 1998; Agichtein and Gravano, 2000; Ravichandran and Hovy, 2002) rely on the abundance of web data to obtain easy patterns and learn such patterns based mostly on lexical information. Rather than analyzing dependency path between entity pair proposed in (Bunescu and Mooney, 2006; Cui et al., 2005), our method analyzes a subtree derived from the dependency structure. Such subtree contains more evidence of the entities' inter-relation than the path in some cases. We propose a new feature obtained from the subtree by using a subtree-mining technique.

In addition, we also make use of the characteristics of Wikipedia to allocate the mentions of entities and further identify their types to help the relation extraction process.

2 Wikipedia's Article Characteristics

Due to the *encyclopedic style*, each Wikipedia article mainly provides information for a specific entity and further mentions other entities related to it. Culotta et al. (2006) defines the entities as *principal entity* and *secondary entity* respectively. We predict only relationships between the principal entity and each mentioned secondary entity that contains a link to its descriptive article.

We put some assumptions in this study: a relationship can be expressed completely in one sentence. Furthermore, a relationship between an entity pair might be expressed with the implication of the principal entity in some cases. Thus, for an article, only sentences containing at least a secondary entity are necessarily analyzed.

An interesting characteristic of Wikipedia is the category hierarchy that is used to classify articles according to their content. Additionally, those articles for famous entities provide summary sections on their right side, which are created by human editors. Finally, the first sentence of an article often defines the principal entity.

3 Proposed Method

Figure 1 delineates our framework for relation extraction. First, Wikipedia articles are processed to remove HTML tags and to extract hyperlinks that point to other Wikipedia articles. Raw text is submitted to a pipeline including a *Sentence Splitter*, a *Tokenizer* and a *Phrase Chunker* supplied by the OpenNLP [1] tool set. The instances of the principal entity and secondary entities are then anchored in the articles. The *Secondary Entity Detector* simply labels the appropriate surface texts of the hyperlinks to other Wikipedia articles, which are proper

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

Proceedings of NAACL HLT 2007, Companion Volume, pages 125–128,
Rochester, NY, April 2007.

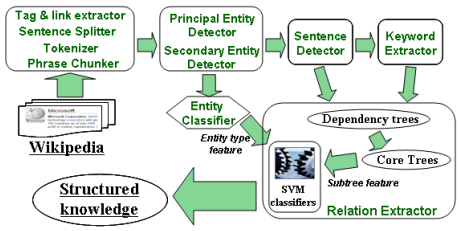


Figure 1: System framework

nouns as secondary entities. The *Principal Entity Detector* will be explained in the following subsection.

After the entities are anchored, sentences that include at least one mention of secondary entities will be selected by a *Sentence Detector*. Each mention of the secondary entities is considered as a relation candidate between the underlying entity and the principal entity. Secondary entities are always explicit, although the principal entity is sometimes implicit in sentences containing no mention.

Keywords that provide clues for each relation label will be identified by a *Keyword Extractor*. Parallely, an *Entity Classifier* module classifies the entities into types. The *Relation Extractor* extracts *subtree feature* from a pair of the principal entity and a mention of secondary entity. It then incorporates *subtree feature* together with *entity type feature* into a feature vector and classifies relations of the entity pair using *SVM-based classifiers*.

3.1 Principal Entity Detector

This module detects all *referring expressions* of the principal entity in an article. All occurrences of identified expressions are labeled as mentions of the principal entity. We adopt (Morton, 2000) to classify the expressions into three types: (1) personal pronoun (2) proper noun (3) common nouns. Based on chunking information, we propose a simple technique to identify a set of referring expressions of the principal entity, denoted as F:

(i) Start with $F = \{\}$.

(ii) Select the first two chunks for F: the *proper chunk* (nounphase with at least one proper noun) of the *article title* and the first proper chunk in the *first sentence* of the article, if any. If F is still empty, stop.

(iii) For each remaining proper chunk p in the article, if p is derived from any expressions selected in (ii), then $F \leftarrow p$. Proper chunk p_1 is derived from proper chunk p_2 if all its proper nouns appear in p_2.

(iv) In the article, select c as the most frequent *subjective pronouns*, find *c'* as its equivalent *objective pronoun* and add them to F.

(v) For each chunk p with the pattern [DT N_1 ... N_k] where DT is a *determiner* and N_k's are a *common nouns*, if p appears more frequently than all the selected pronouns in (iv), then $F \leftarrow p$.

Table 1: Sample extracted referring expressions

Article	Referring expressions	Step
Bill Gates	[NP Bill/NNP Gates/NNP]	(ii)
	[NP William/NNP H./NNP Gates/NNP]	(ii)
	[NP Gates/NNP]	(iii)
	[NP The/DT Gates/NNP]	(iii)
	[NP he/PRP]	(iv)
	[NP him/PRP]	(iv)
Microsoft	[NP Microsoft/NNP]	(ii)
	[NP The/DT Microsoft/NNP Corporation/NNP]	(ii)
	[NP that/DT Microsoft/NNP]	(iii)
	[NP It/PRP]	(iv)
	[NP the/DT company/NN]	(v)
Microsoft Windows	[NP Microsoft/NNP Windows/NNP]	(ii)
	[NP Microsoft/NNP]	(iii)
	[NP Windows/NNP]	(iii)
	[NP the/DT Windows/NNP]	(iii)
	[NP it/PRP]	(iv)

Table 2: List of relations and their keywords

Relation	Keywords
CEO	CEO, chief, executive, officer
Chairmans	chairman
COO	coo, chief, operating, officer
Director	director
Founder	found, founder, founded, establish, form, foundation, open
President	president
Vice chairman	vice, chairman
Birth date	born, bear, birth, birthday
Birth place	born, bear
Foundation	found, establish, form, founded, open, create, formed, established, foundation, founding, cofounder, founder
Location	headquartered, based, locate, headquarter, base, location, situate, located
Product	product, include, release, produce, service, operate, provide, market, manage, development, focus, manufacture, provider, launch, make, sell, introduce, producer, supplier, possess, retailer, design, involve, production, offering, serve, sale, supply
Spouse	marry, wife, married, husband, marriage

Table 1 shows some extracted referring expressions. The third column indicates in which step the expressions are selected. Supported by the nature of Wikipedia, our technique provides better results than those of the coreference tool in LingPipe library [2] and OpenNLP tool set.

3.2 Entity Classifier

Entity type is very useful for relation extraction. For instance, the relation label between a *person* and an *organization* should be *founder*, *chairman*, etc., but cannot be *spouse*, *product*, etc. We first identify *year*, *month* and *date* entities by directly examining their surface text. Types of other entities are identified by classifying their corresponding articles. We develop one SVM-based classifier for each remaining type using the following features: **category feature** (categories collected when tracing from the article upto k level of its category structure), **pronoun feature** (the most frequent *subjective pronoun* in the article) and **singular noun feature** (singular nouns of the first sentence of the article).

3.3 Keyword Extractor

Our hypothesis in this research is that there exist some keywords that provide clues for the relationship between

[2] http://www.alias-i.com/lingpipe/index.html

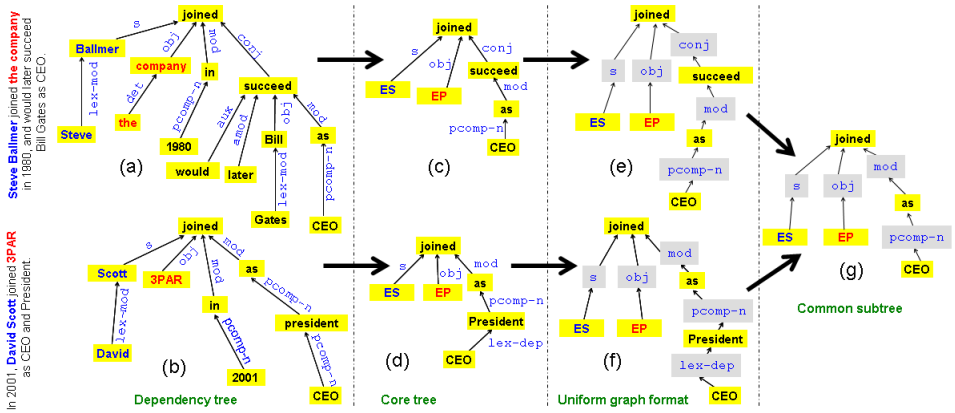


Figure 2: Dependency trees in (a) & (b); core trees with respect to *CEO* relationship in (c) & (d); new representation of the core trees in (e) & (f); common subtree in (g). The red phrase *EP* denotes the principal entity; the blue phrase *ES* denotes the secondary entity.

a pair. For example, to express the *founder* relation, a sentence should contain one keyword such as: *found*, *founder*, *founded*, *co-founders*, or *establish*, etc. We identify such keywords by using a semi-automatic method. First, we automatically extract some true relations from summary sections of Wikipedia articles. Then, we map entities in such relations to those in sentences to build sample sentences for each relationship . *Tf-idf* model is exploited to measure the relevance of words to each relationship for those on the dependency path between the entity pair. Finally, we choose the keywords manually from lists of candidates ranked by relevance score with respect to each relation. Table 2 shows our result selected from ranked lists of total 35,820 keyword candidates using only one hour of human labor.

3.4 Subtree Feature from Dependency Path

In this subsection, we will describe how to obtain efficient features for extracting relation using subtree mining. We extend the idea of Bunescu et al. (Bunescu and Mooney, 2006) suggesting the analysis of dependency path between the entities for extracting relation, in that paths between the secondary entity and the keywords of r will be added to the dependency path between the entities to create a tree. The expanded tree is defined as *core tree* of r because it attempts to capture the clues for r. Steps to extract the core tree C of a relationship r from a sentence s are described as follows:

(i)] Initialize the core tree C as blank.

(ii) Derive the dependency tree D from s.

(iii) Label the group of nodes corresponding to words of secondary entity by an ES node in D.

(iv) If the principal entity appears in s, apply (iii) to replace principal entity with EP. Then extract P_0 as shortest path from ES to EP in D and add $P_0 \rightarrow C$.

(v) For each keyword w of r, extract P_w as the shortest path from ES to node of w and add $P_w \rightarrow C$.

Figures 2c & 2d present exemplary core trees of *CEO* relationship derived from the dependency trees in Figures 2a & 2b. To analyze both words and relations of a core tree uniformly, we transform it into a *uniform graph format* (Figures 2e & 2f) in which core tree words and relations are also represented as graph nodes.

We define a basic element of a relationship r as a key pattern that commonly appears in various core trees of r. As an example, the core trees in Figures 2e & 2f share a common pattern in Figure 2g. Intuitively, this subtree shares the core trees of sentences that express the idea of *"joined the company as CEO"* or *"joined the company and do something as CEO"*.

We denote $T = (V, E)$ as a directed tree, in which V is a set of nodes and E is a set of directed edges. Node y is an ancestor of node x, denoted by $x \prec y$, if $(x,y) \in E$ or $\exists i_1,...,i_k$ ($k \in \mathbb{N}$ and $k \geq 1$) such that $(x,i_1),(i_1,i_2),...,(i_{k-1},i_k),(i_k,y) \in E$. We define that a tree $S = (V_S, E_S)$ is a subtree of T if and only if: (i) $V_S \subset V$, and (ii) $\forall (x,y) \in E_S$, we have $x \prec y$ in T.

We use a subtree as a feature for relation extraction. From a set of training sentences with respect to a relationship r, we derive the core trees. A frequent tree-mining algorithm (Zaki, 2002) is used to generate subtrees from that set of core trees to form the feature space. Each mined subtree corresponds to a value of the feature.

4 Experiments and Evaluations

In this experiment, 5,975 articles are selected, in which 45 articles are for testing and 5,930 articles for training. We apply the framework in Figure 1 on the training articles to extract keywords and select relation candidates. Subsequently, 3,833 positive instances (each contains at least one relation) and 805 negative instances (the ones containing no relation) from the candidates are annotated to train the *Relation Extractor*. Among 39,467

Table 3: Compare our proposed system and baselines

	Precision(%)	Recall(%)	F1(%)
B0	8.70	22.26	12.51
B1	9.88	25.31	14.21
DepTree	29.07	53.86	37.76

Table 4: Result of *Entity Classifier* with various levels (k value) of exploited category structure

Depth k(%)	Accuracy(%)
1	64.0
2	69.5
3	81.0
4	**81.5**
5	79.5
6	77.5
7	77.0
8	78.0
9	75.0
10	74.5

entities collected from all principal and secondary entities, we randomly select 3,300 entities and manually annotate their types for the *Entity Classifier*. Finally, we use 3,100 entities for training and 200 entities for testing.

We develop two baseline systems to evaluate our method, which use bag-of-words model. The second system (B1 in Table 3) works like the *Keyword Extractor* on training instances in that it calculates *tf-idf* scores for words on the dependency path between the entities with respect to each relation. During testing, it accumulates *tf-idf* scores of words on the path and chooses the relation label that gives the highest score for the entity pair. The only difference between the two baseline systems is that the first one (B0 in Table 3) focuses on all the words between the entities in sentence text, not dependency path.

In our experiments, dependency graphs are obtained by Minipar parser (Lin, 1998), classifiers are trained by SVM Light (Joachims, 1999) with 2nd- order polynomial kernel, subtrees are mined by FREQT [3] tree miner.

On the basis of preliminary experiments, we report the performance of our system compared with those of baseline systems in Table 3. The result shows that our proposed method gives a substantial improvement over the baselines. Although the recall is quite adequate, precision is low. Data analysis reveals that although the mined subtrees capture key features for relationships, they also generate many irrelevant features which degrade the performance. It is necessary to carry out feature selection step for subtree feature. One more reason of the poor precision is that our system suffers from the error accumulation in a long pipeline of *entity detection*, *entity classification*, *dependency parsing* and *relation classification*.

Table 4 shows the effectiveness of different values of k parameter in *Entity Classifier*. The classifier works best when we trace *four levels* on category system. An interesting fact is that Wikipedia can be used as an external knowledge source for Named Entity Recognition.

[3] http://chasen.org/taku/software/freqt/

5 Conclusions and Future Works

We have presented a method to extract relations between entities from Wikipedia articles by incorporating information from the Wikipedia structure and by the analysis of Wikipedia text. The key features of our method include: (1) an algorithm to build the core syntactic tree that reflects the relation between a given entity pair more accurately; (2) the use of a tree-mining algorithm to identify the basic elements of syntactic structure of sentences for relationships; (3) method to make use of the nature of Wikipedia for entity allocation and entity classification.

References

E. Agichtein and L. Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *the 5th ACM International Conference on Digital Libraries*.

S. Brin. 1998. Extracting patterns and relations from the world wide web. In *Proceedings of the 1998 International Workshop on the Web and Databases*, pages 172–183.

R.C. Bunescu and R.J. Mooney. 2006. Extracting relations from text: From word sequences to dependency paths. In Anne Kao and Steve Poteet, editors, *Text Mining and Natural Language Processing*.

H. Cui, R. Sun, K. Li, M.-Y. Kan, and T.-S. Chua. 2005. Question answering passage retrieval using dependency relations. In *Proceedings of SIGIR*.

A. Culotta, A. McCallum, and J. Betz. 2006. Integrating probabilistic extraction models and data mining to discover relations and patterns in text. In *Proceedings of the HLT-NAACL-2006*.

T. Joachims. 1999. Making large-scale svm learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT-Press.

D. Lin. 1998. Dependency-based evaluation of minipar. In *Proceedings of the Workshop on the Evaluation of Parsing Systems,* 1st *International Conference on Language Resources and Evaluation*.

T. Morton. 2000. Coreference for nlp applications. In *Proceedings of the ACL-2000*.

D. Ravichandran and E. Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the ACL-2002*, pages 41–47.

M.J. Zaki. 2002. Efficiently mining frequent trees in a forest. In *Proceedings of 8th ACM SIGKDD*.

ADVANCES IN THE CMU/INTERACT ARABIC GALE TRANSCRIPTION SYSTEM

Mohamed Noamany, Thomas Schaaf[*]*, Tanja Schultz*

InterACT, Language Technologies Institute, Carnegie Mellon University
Pittsburgh, PA 15213
{mfn,tschaaf,tanja@cs.cmu.edu}

ABSTRACT

This paper describes the CMU/InterACT effort in developing an Arabic Automatic Speech Recognition (ASR) system for broadcast news and conversations within the GALE 2006 evaluation. Through the span of 9 month in preparation for this evaluation we improved our system by 40% relative compared to our legacy system. These improvements have been achieved by various steps, such as developing a vowelized system, combining this system with a non-vowelized one, harvesting transcripts of TV shows from the web for slightly supervised training of acoustic models, as well as language model adaptation, and finally fine-tuning the overall ASR system.

Index Terms— Speech recognition, Vowelization, GALE, Arabic, Slightly supervised training, web data.

1. INTRODUCTION

The goal of the GALE (Global Autonomous Language Exploitation) program is to develop and apply computer software technologies to absorb, analyze and interpret huge volumes of speech and text in multiple languages and make them available in English. In a long run this requires to combine techniques from text summarization, information retrieval, machine translation, and automatic speech recognition. NIST will perform regular evaluations and the first evaluation took place recently. This paper describes improvements in the CMU Modern Standard Arabic (MSA) system through the span of 9 months in preparation for this evaluation.

One of the language characteristics and challenges of Arabic is that some vowels are omitted in the written form. These vowels carry grammatical case information and may change the meaning of a word. Modeling the vowels in the pronunciation dictionary was found to give improvements over un-vowelized pronunciations [4]. In this paper we achieved another significant improvement by combining a vowelized with a non-vowelized system. Furthermore, we got gains by collecting and utilizing web transcripts from TV show, which include broadcast conversations.

2. SYSTEM DESCRIPTION

Our MSA speech recognition system is based on the Janus Recognition Toolkit JRTk [9] and the IBIS decoder [10].

Before decoding the audio, an automatic segmentation step and a speaker clustering step is performed. The segmentation step aims at excluding those segments that contain no speech, such as music or background noise. The remaining segments are clustered into speaker clusters such that all adaptation and normalization steps can be processed on clusters as batches.

From the incoming 16 kHz audio signal we extract for each segment power spectral features using a FFT with a 10ms frame-shift and a 16ms Hamming window. From these we compute 13 Mel-Frequency Cepstral Coefficients (MFCC) per frame and perform a cepstral mean as well as variance normalization on a cluster basis. To incorporate dynamic features we concatenate 15 adjacent MFCC frames (±7) and project these 195 dimensional features into a 42 dimensional space using a transform found by linear discriminate analysis (LDA). We use the context-dependent codebooks as classes for finding the LDA transform [2]. On top of the LDA we apply a single maximum likelihood trained Semi-Tied-Covariance (STC) matrix.

The general decoding setup employs a first pass in which a speaker independent acoustic model without vocal tract length normalization (VTLN) and no adaptation is used. The hypotheses of a cluster from the first pass are then used to estimate the VTLN warping factors to warp the power spectrum using the maximum likelihood approach described in [8]. After the VTLN factors are found, the same hypotheses are considered to estimate a feature space adaptation (FSA) using a constrained MLLR (CMLLR) transform. Then a model space adaptation is performed using maximum likelihood linear regression with multiple regression classes. The regression classes are found through clustering of the Gaussians in the acoustic model. The second pass decoding uses a speaker adaptive trained

[*] Now with Toshiba Research Europe Ltd, Cambridge, United Kingdom

Proceedings of NAACL HLT 2007, Companion Volume, pages 129–132,
Rochester, NY, April 2007.

acoustic model, in which the adaptation was performed using a single CMLLR transform per speaker.
For the non-vowelized system, we applied a grapheme-to-phoneme approach to automatically generate the pronunciation dictionary. For the vowelized system we used the same phoneme set as in the non-vowelized system but extended it with the 3 short vowels, which do not appear in the writing system. Both systems are 2-pass system as described above and employ Cepstral Mean Normalization (CMN), MLLR, Semi-tied covariance (STC), and Feature space adaptation (FSA).

For the development of context dependent acoustic models we applied an entropy-based polyphone decision tree clustering process using context questions of maximum width ±2, resulting in shared quin-phones. In addition we included word-boundary tags into the pronunciation dictionary, which can be asked for in the decision tree can ask for word-boundary tags. The non-vowelized system uses 4000 phonetically-tied quin-phones with a total of 305,000 Gaussians. The non-vowelized system has 5000 codebooks with a total of 308,000 Gaussians.

In total we used 190 hours for acoustic training. These consist of 40 hours Broadcast news (BN) from manually transcribed FBIS data, 50 hours BN LDC-TDT4 selected from 85 hours using a slightly supervised approach as described in [3], and 30 hours Broadcast conversation (BC) recorded from Al-jazeera TV, and 70 hours (40hrs BN, 30hrs BC) from LDC-GALE data. For quality reasons we removed some of the most recent GALE data from acoustic model training.

4. LANGUAGE MODELING

The Arabic Giga word corpus distributed by LDC is currently the major Arabic text resource for language modeling. Since this corpus only covers broadcast news, we spidered the web to cover broadcast conversational data. We found transcripts for Arabic talk shows on the Al-jazeera web site www.al-jazeera.net and collected all data available from 1998 to 2005. We excluded all material from 2006 to comply the evaluation rules which prohibit the use of any data starting February 2006. In addition to the mentioned data we collected BN data from the following source: Al-Akhbar (Egyptian daily newspaper 08/2000 to 12/2005) and Akhbar Elyom (Egyptian weekly newspaper 08/2000 to 12/2005). Furthermore, we used unsupervised training transcripts from 750 hours BN created and shared by IBM.

For language modeling building we used the SRILM tool kit from SRI [5]. Since we have 2 kinds of data, Broadcast News and Conversation, we built various individual 4-grams language models. 11 models were then interpolated to create one language model. The interpolation weights were selected based on a held out data set from BN and BC sources. We found that the data from Al-jazeera (both BN & BC) has the highest weight comparing to other sources. The resulting final language model uses a total number of n-grams is 126M and a vocabulary of 219k words. The perplexity of the language model is 212 on a test set containing BC and BN data.

5. TV WEB TRANSCRIPTS

Most of our acoustic and language model training data comes from broadcast news. However, since GALE targets broadcast news as well as conversations we looked for an effective method to increase the training data for Arabic BC. We made use of the fact that some Arabic TV stations place transcripts for their program on the web. These transcripts lack time stamp but include acceptable quality of the transcription. However, one challenge is that the transcriptions are not complete in that they do not include transcripts of commercials or any news break that may interrupt the show. In total we recorded 50 hours of Broadcast conversation shows from Al-jazeera and used them in our acoustic model and language model training by performing the following procedures:

- We manually selected shows from Al-jazeera TV
- We used a scheduler to automatically start the recording of the selected shows.
- We spidered the web to collect corresponding show transcripts from their web site www.aljazeera.net.
- We automatically processed the transcripts to convert the html files to text, convert numbers to words and remove any non-Arabic words in the shows.
- We added these shows to our LM data with high weight, built a biased LM, and used this LM to decode the recorded shows.
- We aligned the reference (transcripts without time stamps) with the decoder output that may contain speech recognition errors.
- We selected only the portions that are correct; we did not select any portion with number of words less than 3 correct consecutive words.
- Based on the above criteria we finally selected 30 hours out of the total 40 hours recordings.
- We clustered utterances based on BIC criteria approach described in [7].

As a result, we managed to project the time stamp in the original transcript such that it can be used for training. Using these 30 hours of data resulted in a 7% relative improvement on RT04. Since RT04 is broadcast news, we expect even higher gains on broadcast conversational data. It is worth mentioning that we

applied the same slightly supervised approach to the TDT4 data which is a low quality quick transcription. We selected 50 out of 80 hours and achieved an improvement of 12% relative. The gain was higher since at the time of these experiments we had only 40 hours of training from FBIS data, therefore more than doubled the amount of training data by adding TDT4.

6. NON-VOWELIZED SYSTEM

Arabic spelling is mostly phonemic; there is a close letter-to-sound correspondence. We used a grapheme-to-phoneme approach similar to [1]. Our phoneme set contains 37 phonemes plus three special phonemes for silence, non-speech events, and non-verbal effects, such as hesitation.

We preprocessed the text by mapping the 3 shapes of the grapheme for glottal stops to one shape at the beginning of the word since these are frequently miss-transcribed. This preprocessing step leads to 20% reduction in perplexity of our language model and 0.9% improvements in the final WER performance on RT04. Preprocessing of this kind appears to be appropriate since the target of the project is not transcription but speech translation and the translation community applies the same pre-processing. We used a vocabulary of 220K words selected by including all words appearing in the acoustic transcripts and the most frequent words occurring in the LM. The OOV rate is 1.7% on RT04. Table 1 shows the performance of our Speaker-Independent (SI) and Speaker-Adaptive (SA) non-vowelized system on the RT04 set.

Table 1: Non-vowelized System Results

System		WER on RT04 (%)
Non-Vowelized	(SI)	25.3
Non-Vowelized	(SA)	20.8

7. VOWELIZED SYSTEM

Written MSA lacks vowels, thus native speakers add them during reading. Vowels are written only in children books or traditional religious books. To restore vowels for a 129K vocabulary [4], we performed the following steps:

- Buckwalter morphological analyzer (BMA) (found 106K out of 129K entries).
- If a word is not vowelized by the analyzer, we check for its vowelization in the LDC Arabic Tree-Bank (additional 5k entries found).
- If the word did not appear in any of those, we used the written non-vowelized word form.

In total 11k entries could not be resolved by either the BMA or the Treebank.

This vowelization step resulted in 559,035 pronunciations for the 129k words in our vocabulary, i.e. we have on average 5 pronunciations per word. To reduce the number of pronunciation variants we performed a forced alignment and excluded pronunciations which did not occur in the training corpus. This results in 407,754 pronunciations, which is a relative reduction of about 27%. For system training we used the same vocabulary and applied the same training procedure as in the non-vowelized system for acoustic model training.

As Table 2 shows, we achieved a very good gain of 1.3% absolute on the SI pass and 1.5% on the SA pass, both benchmarked on RT04 (compare Table 1). We envision to seeing even higher improvements after estimating and applying probability priors to multiple pronunciation and after vowelizing the remainder 11k words that had not been covered by BMA or the Tree-Bank.

Table2: Vowelized System Results

System		WER on RT04 (%)
Vowelized	(SI)	24.0
Vowelized	(SA)	19.3

8. COMBINING VOWELIZED & NON-VOWELIZED SYSTEM

After seeing significant improvements by vowelization, we investigated the performance gain through cross-adapting the vowelized system with the non-vowelized system. The vowelized system cross adapted with the SA non-vowelized gave us 1.3 over the vowelized system adapted on the SI vowelized system. We used a 3-pass decoding strategy, in which the first pass uses the speaker independent (SI) vowelized system, the second pass uses the speaker adaptive (SA) non-vowelized system, and the third, final pass, uses the speaker adaptive vowelized system. Some challenges for the cross-adaptations had to be overcomed, for instance to cross adapt the non-vowelized system on the vowelized system, we had to remove the vowels to have a non-vowelized transcript. Since the phoneme set of the non-vowelized system is a subset of the phoneme set of the vowelized system, we could simply exclude the vowel phonemes from the vowelized system. Furthermore, the search vocabulary is the same and so is the language model.

The main changes are the pronunciation dictionary and the decision tree. We tried different combination schemes, e.g. by starting with the non-vowelized system, then the vowelized, and then the non-vowelized but found that none outperforms the combination reported here in terms of WER. In addition starting with the non-vowelized SI pass is much faster than the vowelized SI system (4.5RT compared to 9RT).

Table 3: Non-vowelized & vowelized System Combination

System	WER on RT04 (%)
Vowelized (SI)	24.0
Non-Vowelized (SA)	19.9
Vowelized (SA)	18.3

9. ACOUSTIC MODEL PARAMETER TUNING

We started our legacy system with 40 hours and until it reached 90 hours we were using the same number of codebooks (3000) and same number of Gaussians (64) per codebook. With the increase of training data from 90 hours to 190 hours we investigated the effect of increasing the number of codebooks and Gaussians. Also, we were using merge and split training (MAS) and STC only for the adapted pass; we furthermore investigated the effect of using it for the SI pass. We found that using MAS & STC on the SI pass gave us a gain of 5% relative on the SI pass. In addition we found that the ideal number of codebooks is 5000 for the non-vowelized system resulting in a gain of 5.3% relative on the SI pass. We expect to see further gains on the SA pass. Table 4 summarizes the system performance using different parameter sizes and training schemes.

Table 4: System Performance vs.Model Size

#codebooks	MAS	#Gausians	Voc	System	WER(%)
3K	-	64K	129	Non-vow(NV)	29.6
3K	***Mas***	64K	129	NV	28.3
5K	Mas	64K	129	NV	27.9
5K	Mas	***100K***	129	NV	27.6
5K	Mas	100K	200	nv+***tv TRANS***	26.3
3K	Mas	100K	200	***vow***+ tvTRANS	24.0

10. SYSTEM EVOLUTION

Table 5 shows the gains we achieved at major milestone stages while building the system. The key improvements are due to adding data collected from the web, Vowelization, and combining the vowelized and non-vowelized systems. Tuning the acoustic models parameters gave us a good gain and finally the interpolation of different language model for different sources gave additional improvements. The real-time behavior of the system improved from 20RT to 10 RT while loosing only 0.2% which is in acceptable trade-off. Recently, we gained 3.5% relative applying discriminative training (MMIE).

11. CONCLUSION

We presented the CMU 2006 GALE ASR Arabic system. It can be seen that we achieved 40% improvements over our legacy system.

Table 5: System Progress WER (%)

LEGACY SYSTEM	32.7
STC+VTLN	30.1
SPEED FROM 20RT TO 10RT	30.3
FROM 3 TO 4GM+BETTER SEGMENTATION	28.4
TDT4 TRANSCIPTS SELECTION REFINEMENT	26.3
CLUSTERING REFINEMENT & RETRAINING	25.5
MORE LM DATA +INTERPOLATING 11 LMS	24.2
ADDITION Q3 OF LDC DATA	23.6
ACOUSTIC MODEL PARAMETER TUNING	20.7
MMIE	20.0
COMBINED SYSTEMS (VOW+NON-VOW)	18.3

We combined a vowelized and a non-vowelized system and achieved 4.0% relative over the vowelized system. Also, we managed to use TV web transcript as a method to cover the shortage of training data specially the broadcast conversation. Currently, we are exploring more on the vowelized system by adding weights to different multiple pronunciations and adding vowelization to words not covered by the morphological analyzer or the tree-bank.

12. ACKNOWLEGMENTS

We also would like to thank Qin Jin for applying her automatic clustering techniques to the web data.

13. REFERENCES

[1] J. Billa, M. Noamany, A. Srivastava, D. Liu, R. Stone, J. Xu, J. Makhoul, and F. Kubala, "Audio Indexing of Arabic Broadcast News", International Conference on Acoustics, Speech, and Signal Processing (ICASSP), May 2002.

[2] H. Yu, Y-C. Tam, T. Schaaf, S. Stüker, Q. Jin, M. Noamany, and T. Schultz "The ISL RT04 Mandarin Broadcast News Evaluation System", EARS Rich Transcription workshop, Palisades, NY, 2004.

[3] L. Nguyen et al., "Light supervision in acoustic model training," ICASSP , Montreal, QC, Canada, May 2004.

[4] M. Afify et al.,"Recent progress in Arabic broadcast news transcription at BBN", In INTERSPEECH-2005.

[5] A. Stolcke SRILM- An Extensible Language Modeling ToolKit ICSLP. 2002, Denver, Colorado.

[6] T.Buckwalter, "Issues in Arabic Orthography and morphology Analysis", COLING 2004, Geneva, 2004.

[7] Q. Jin, T. Schultz, "Speaker segmentation and clustering in meetings", ICSLP, 2004.

[8] P. Zhan, M. Westphal, "Speaker Normalization Based On Frequency Warping", ICASSP 1997, Munich, Germany.

[9] M. Finke, et al., "*The Karlsruhe Verbmobil Speech Recognition Engine*," International Conference on Acoustics, Speech, and Signal Processing, ICASSP,1997.

[10] H. Soltau, et al., "A One Pass-Decoder Based On Polymorphic Linguistic Context", ASRU 2001, Trento, Italy, 2001

An integrated architecture for speech-input multi-target machine translation

Alicia Pérez, M. Inés Torres
Dep. of Electricity and Electronics
University of the Basque Country
manes@we.lc.ehu.es

M. Teresa González, Francisco Casacuberta
Dep. of Information Systems and Computation
Technical University of Valencia
fcn@dsic.upv.es

Abstract

The aim of this work is to show the ability of finite-state transducers to simultaneously translate speech into multiple languages. Our proposal deals with an extension of stochastic finite-state transducers that can produce more than one output at the same time. These kind of devices offer great versatility for the integration with other finite-state devices such as acoustic models in order to produce a speech translation system. This proposal has been evaluated in a practical situation, and its results have been compared with those obtained using a standard mono-target speech transducer.

1 Introduction

Finite-state models constitute an important framework both in syntactic pattern recognition and in language processing. Specifically, stochastic finite-state transducers (SFSTs) have proved to be useful for machine translation tasks within restricted domains; they usually offer high speed during the decoding step and they provide competitive results in terms of error rates (Mohri et al., 2002). Moreover, SFSTs have proved to be versatile models, which can be easily integrated with other finite-state models (Pereira and Riley, 1997).

The article (Casacuberta and Vidal, 2004) explored an automatic method to learn an SFST from a bilingual set of samples for machine translation purposes, the so-called GIATI (*Grammar Inference and Alignments for Transducers Inference*). It described how to learn both the structural and the probabilistic components of an SFST making use of underlying alignment models.

A multi-target SFST is a generalization of standard SFSTs, in such a way that every input string in the source language results in a tuple of output strings each being associated to a different target language. An extension of GIATI that allowed to infer a multi-target SFST from a multilingual corpus was proposed in (González and Casacuberta, 2006). A syntactic variant of this method (denoted as GIAMTI) has been used in this work in order to infer the models from training samples as it is summarized in section 3.

On the other hand, speech translation has been already carried out by integrating acoustic models into a SFST (Casacuberta et al., 2004). Our main goal in this work is to extend and assess these methodologies to accomplish spoken language multi-target translation. Section 2 deals with this proposal by presenting a new integrated architecture for speech-input multi-target translation. Under this approach spoken language can be simultaneously decoded and translated into m languages using a unique network. In section 4, the performance of the system has been experimentally evaluated over a trilingual task which aims to translate TV weather forecast into two languages at the same time.

2 An integrated architecture for speech-input multi-target translation

The classical architecture for spoken language multi-target translation involves a speech recogni-

Proceedings of NAACL HLT 2007, Companion Volume, pages 133–136,
Rochester, NY, April 2007.

tion system in a serial architecture with m decoupled text-to-text translators. Thus, the whole process involves $m+1$ searching stages, a first one for the speech signal transcription into the source language text string, and further m for the source language translation into the m target languages. If we replaced the m translators by the multi-target SFST, the problem would be reduced to 2 searching stages. Nevertheless, in this paper we propose a natural way for acoustic models to be integrated in the same network. As a result, the input speech-signal can be simultaneously decoded and translated into m target languages just in a single searching stage.

Given the acoustic representation ($\mathbf{x}$) of a speech signal, the goal of multi-target speech translation is to find the most likely m target strings ($\mathbf{t}^m$); that is, one string ($\mathbf{t}_i$) per target language involved ($i \in \{1, \ldots, m\}$). This approach is summarized in eq. (1), where the hidden variable $\mathbf{s}$ can be interpreted as the transcription of the speech signal:

$$\widehat{\mathbf{t}^m} = \arg\max_{\mathbf{t}^m} P(\mathbf{t}^m|\mathbf{x}) = \arg\max_{\mathbf{t}^m} \sum_{\mathbf{s}} P(\mathbf{t}^m, \mathbf{s}|\mathbf{x}) \tag{1}$$

Making use of Bayes' rule, the former expression turns into:

$$\widehat{\mathbf{t}^m} = \arg\max_{\mathbf{t}^m} \sum_{s} P(\mathbf{t}^m, \mathbf{s}) P(\mathbf{x}|\mathbf{t}^m, \mathbf{s}) \tag{2}$$

Empirically, there is no loss of generality if we assume that the acoustic signal representation depends only on the source string: i.e., that $P(\mathbf{x}|\mathbf{t}^m, \mathbf{s})$ is independent of $\mathbf{t}^m$. In this sense, eq. (2) can be rewritten as:

$$\widehat{\mathbf{t}^m} = \arg\max_{\mathbf{t}^m} \sum_{s} P(\mathbf{t}^m, \mathbf{s}) P(\mathbf{x}|\mathbf{s}) \tag{3}$$

Equation (3) combines a standard acoustic model, $P(\mathbf{x}|\mathbf{s})$, and a multi-target translation model, $P(\mathbf{t}^m, \mathbf{s})$, both of whom can be integrated on the fly during the searching routine. Nevertheless, the outer maximization is computationally very expensive to search for the optimal tuple of target strings $\mathbf{t}^m$ in an effective way. Thus we make use of the so called Viterbi approximation, which finds the best path.

3 Inference

Given a multilingual corpus, that is, a finite set of multilingual samples $(\mathbf{s}, \mathbf{t_1}, \ldots, \mathbf{t_m}) \in \Sigma^* \times \Delta_1^* \times \cdots \times \Delta_m^*$, where $\mathbf{t_i}$ denotes the translation of the source sentence $\mathbf{s}$ (formed by words of the input vocabulary Σ) into the i-th target language, which, in its turn, has a vocabulary Δ_i, the GIAMTI method can be outlined as follows:

1. Each multilingual sample is transformed into a single string from an *extended vocabulary* ($\Gamma \subseteq \Sigma \times \Delta_1^* \times \cdots \times \Delta_m^*$) using a *labelling function* ($\mathcal{L}^m$). This transformation searches an adequate monotonous segmentation for each of the m source-target language pairs. A monotonous segmentation copes with monotonous alignments, that is, $j < k \Rightarrow a_j < a_k$ following the notation of (Brown et al., 1993). Each source word is then joined with a target phrase of each language as the corresponding segmentation suggests. Each *extended symbol* consists of a word from the source language plus zero or more words from each target language.

2. Once the set of multilingual samples has been converted into a set of single extended strings ($\mathbf{z} \in \Gamma^*$), a stochastic regular grammar can be inferred.

3. The extended symbols associated with the transitions of the automaton are transformed into one input word and m output phrases ($w/\tilde{p}_1/\ldots/\tilde{p}_m$) by the inverse labeling function ($\mathcal{L}^{-m}$), leading to the required transducer.

In this work, the first step of the algorithm (as described above), which is the one that handles the alignment and segmentation routines, relies on statistical alignments obtained with GIZA++ (Och, 2000). The second step was implemented using our own language modeling toolkit, which learns stochastic k-testable in the string-sense grammars (Torres and Varona, 2001), and allows for back-off smoothing.

4 Experimental results

4.1 Task and corpus

We have implemented a highly practical application that could be used to translate on-line TV weather forecasts into several languages, taking the speech of the presenter as the input and producing as output text-strings, or sub-titles, in several languages. For

this purpose, we used the corpus METEUS (see Table 1) which consists of a set of trilingual sentences, in English, Spanish and Basque, as extracted from weather forecast reports that had been published on the Internet. Basque language is a minority language, spoken in a small area of Europe and also within some small American communities (such as that in Boise, Idaho). In the Basque Country it has an official status along with Spanish. However both languages differs greatly in syntax and in semantics. The differences in the size of the vocabulary (see Table 1), for instance, are due to the agglutinative nature of the Basque language.

With regard to the speech test, the input consisted of the speech signal recorded by 36 speakers, each one reading out 50 sentences from the test-set in Table 1. That is, each sentence was read out by at least three speakers. The input speech resulted in approximately 3.50 hours of audio signal. Needless to say, the application that we envisage has to be speaker-independent if it is to be realistic.

		Spanish	Basque	English
Training	Sentences	14,615		
	Different Sent.	7,225	7,523	6,634
	Words	191,156	187,462	195,627
	Vocabulary	702	1,147	498
	Average Length	13.0	12.8	13.3
Test	Different Sent.	500		
	Words	8,706	8,274	9,150
	Average Length	17.4	16.5	18.3
	Perplexity (3grams)	4.8	6.7	5.8

Table 1: Main features of the METEUS corpus.

4.2 System evaluation

The experimental setup was as follows: the multi-target SFST was learned from the training set in Table 1 using the GIAMTI algorithm described in section 1; then, the speech test was translated, and the output provided by the system in each language was compared to the corresponding reference sentence. Additionally, two mono-target SFST were inferred from the same training set with their outputs for the aforementioned test to be taken as baseline.

4.2.1 Computational cost

The expected searching time and the amount of memory that needs to be allocated for a given model are two key parameters to bear in mind in speech-input machine translation applications. These values can be objectively measured based on the size and on the average branching factor of the model displayed in Table 2.

	multi-target	mono-target	
		S2B	S2E
Nodes	52,074	35,034	20,148
Edges	163,146	115,526	69,690
Braching factor	3.30	3.13	3.46

Table 2: Features of multi-target model and the two decoupled mono-target models (one for Spanish to Basque translation, referred to as S2B, and the second for Spanish to English, S2E).

Adding the states and the edges up for the two mono-target SFSTs that take part in the decoupled architecture (see Table 2), we conclude that the decoupled model needs a total of $185,216$ edges to be allocated in memory, which represents an increment of 13% in memory-space with respect to the multi-target model.

On the other hand, the multi-target approach offers a slightly smaller branching factor than each mono-target approach. As a result, fewer paths have to be explored with the multi-target approach than with the decoupled one, which means that searching for a translation can be faster. In fact, experimental results in Table 3 show that the mono-target architecture works $\%11$ more slowly than the multi-target one.

	multi-target	mono-target		
		S2B	S2E	S2B+S2E
Time (s)	30,514	24,398	9,501	33,899

Table 3: Time needed to translate the speech-test into two languages.

Summarizing, in terms of computational cost (space and time), a multi-target SFST performs better than the mono-target decoupled system.

4.2.2 Performance

So far, the capability of the systems have been assessed in terms of time and spatial costs. However, the quality of the translations they provide is, doubtless, the most relevant evaluation criterion. In order to assess the performance of the system in a quantitative manner, the following evaluation parameters

were computed for each scenario: *bilingual evaluation under study* (BLEU), *position independent error rate* (PER) and *word error rate* (WER).

As can be derived from the Speech-input translation results shown in Table 4, slightly better results are obtained with the classical mono-target SFSTs, compared with the multi-target approach. From Spanish into English the improvement is around 3.4% but from Spanish into Basque, multi-target approach works better with an improvement of a 0.8%.

	multi-target		**mono-target**	
	S2B	**S2E**	**S2B**	**S2E**
BLEU	39.5	59.0	39.2	61.1
PER	42.2	25.3	41.5	23.6
WER	51.5	33.9	50.5	31.9

Table 4: Speech-input translation results for Spanish into Basque (S2B) and Spanish into English (S2E) using a multi-target SFST or two mono-target SFSTs.

The process of speech signal decoding is itself introducing some errors. In an attempt to measure these errors, the text transcription of the recognized input signal was extracted and compared to the input reference in terms of WER as shown in Table 5.

	multi-target	**mono-target**	
		S2B	**S2E**
WER	10.7	9.3	9.1

Table 5: Spanish speech decoding results for the multi-target SFST and the two mono target SFSTs.

5 Concluding remarks and further work

A fully embedded architecture that integrates the acoustic model into the multi-target translation model for multiple speech translation has been proposed. Due to the finite-state nature of this model, the speech translation engine is based on a Viterbi-like algorithm. The most significant feature of this approach is its ability to carry out both the recognition and the translation into multiple languages integrated in a unique model.

In contrast to the classical decoupled systems, multi-target SFSTs enable the translation from one source language simultaneously into several target languages with lower computational costs (in terms of space and time) and comparable qualitative results.

In future work we intend to make a deeper study on the performance of the multi-target system as the amount of targets increase, since the amount of parameters to be estimated also increases.

Acknowledgements

This work has been partially supported by the University of the Basque Country and by the Spanish CICYT under grants 9/UPV 00224.310-15900/2004 and TIC2003-08681-C02-02 respectively.

References

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Francisco Casacuberta and Enrique Vidal. 2004. Machine translation with inferred stochastic finite-state transducers. *Computational Linguistics*, 30(2):205–225.

F. Casacuberta, H. Ney, F. J. Och, E. Vidal, J. M. Vilar, S. Barrachina, I. García-Varea, D. Llorens, C. Martínez, S. Molau, F. Nevado, M. Pastor, D. Picó, A. Sanchis, and C. Tillmann. 2004. Some approaches to statistical and finite-state speech-to-speech translation. *Computer Speech and Language*, 18:25–47, January.

M. Teresa González and Francisco Casacuberta. 2006. Multi-Target Machine Translation using Finite-State Transducers. In *Proceedings of TC-Star Speech to Speech Translation Workshop*, pages 105–110.

Mehryar Mohri, Fernando Pereira, and Michael Riley. 2002. Weighted finite-state transducers in speech recognition. *Computer, Speech and Language*, 16(1):69–88, January.

Franz J. Och. 2000. GIZA++: Training of statistical translation models.

Fernando C.N. Pereira and Michael D. Riley. 1997. Speech Recognition by Composition of Weighted Finite Automata. In Emmanuel Roche and Yves Schabes, editors, *Finite-State Language Processing*, Language, Speech and Communication series, pages 431–453. The MIT Press, Cambridge, Massachusetts.

M. Inés Torres and Amparo Varona. 2001. k-tss language models in speech recognition systems. *Computer Speech and Language*, 15(2):127–149.

Analysis and System Combination of Phrase- and N-gram-based Statistical Machine Translation Systems

Marta R. Costa-jussà[1], Josep M. Crego[1], David Vilar[2]
José A. R. Fonollosa[1], José B. Mariño[1] and **Hermann Ney[2]**
[1]TALP Research Center (UPC), Barcelona 08034, Spain
{mruiz,jmcrego,adrian,canton}@gps.tsc.upc.edu
[2]RWTH Aachen University, Aachen D-52056, Germany
{vilar,ney}@i6.informatik.rwth-aachen.de

Abstract

In the framework of the TC-STAR project, we analyze and propose a combination of two Statistical Machine Translation systems: a phrase-based and an N-gram-based one. The exhaustive analysis includes a comparison of the translation models in terms of efficiency (number of translation units used in the search and computational time) and an examination of the errors in each system's output. Additionally, we combine both systems, showing accuracy improvements.

1 Introduction

Statistical machine translation (SMT) has evolved from the initial word-based translation models to more advanced models that take the context surrounding the words into account. The so-called phrase-based and N-gram-based models are two examples of these approaches (Zens and Ney, 2004; Mariño et al., 2006).

In current state-of-the-art SMT systems, the phrase-based or the N-gram-based models are usually the main features in a log-linear framework, reminiscent of the maximum entropy modeling approach.

Two basic issues differentiate the N-gram-based system from the phrase-based one: the training data is sequentially segmented into bilingual units; and the probability of these units is estimated as a bilingual N-gram language model. In the phrase-based model, no monotonicity restriction is imposed on the segmentation and the probabilities are normally estimated simply by relative frequencies.

This paper extends the analysis of both systems performed in (Crego et al., 2005a) by additionally performing a manual error analysis of both systems, which were the ones used by UPC and RWTH in the last TC-STAR evaluation.

Furthermore, we will propose a way to combine both systems in order to improve the quality of translations.

Experiments combining several kinds of MT systems have been presented in (Matusov et al., 2006), based only on the single best output of each system. Recently, a more straightforward approach of both systems has been performed in (Costa-jussà et al., 2006) which simply selects, for each sentence, one of the provided hypotheses.

This paper is organized as follows. In section 2, we briefly describe the phrase and the N-gram-based baseline systems. In the next section we present the evaluation framework. In Section 4 we report a structural comparison performed for both systems and, afterwards, in Section 5, we analyze the errors of both systems. Finally, in the last two sections we rescore and combine both systems, and the obtained results are discussed.

2 Baseline Systems

2.1 Phrase-based System

The basic idea of phrase-based translation is to segment the given source sentence into units (here called phrases), then translate each phrase and finally compose the target sentence from these phrase translations.

In order to train these phrase-based models, an alignment between the source and target training sentences is found by using the standard IBM models in both directions (source-to-target and target-to-source) and combining the two obtained alignments. Given this alignment an extraction of contiguous phrases is carried out, specifically we extract all phrases that fulfill the following restrictions: all source (target) words within the phrase are aligned only to target (source) words within the phrase.

The probability of these phrases is normally estimated by relative frequencies, normally in both directions, which are then combined in a log-linear way.

Proceedings of NAACL HLT 2007, Companion Volume, pages 137–140,
Rochester, NY, April 2007.

2.2 N-gram-based System

In contrast with standard phrase-based approaches, the N-gram translation model uses *tuples* as bilingual units whose probabilities are estimated as an N-gram language model (Mariño et al., 2006). This model approximates the joint probability between the source and target languages by using N-grams.

Given a word alignment, tuples define a unique and monotonic segmentation of each bilingual sentence, building up a much smaller set of units than with phrases and allowing N-gram estimation to account for the history of the translation process (Mariño et al., 2006).

2.3 Feature functions

Both baseline systems are combined in a log-linear way with several additional feature functions: a target language model, a forward and a backward lexicon model and a word bonus are common features for both systems. The phrase-based system also introduces a phrase bonus model.

3 Evaluation framework

The translation models presented so far were the ones used by UPC and RWTH in the second evaluation campaign of the TC-STAR project. The goal of this project is to build a speech-to-speech translation system that can deal with real life data.

The corpus consists of the official version of the speeches held in the European Parliament Plenary Sessions (EPPS), as available on the web page of the European Parliament. Table 1 shows some statistics.

The following tools have been used for building both systems: Word alignments were computed using GIZA++ (Och, 2003), language models were estimated using the SRILM toolkit (Stolcke, 2002), decoding was carried out by the free available MARIE decoder (Crego et al., 2005b) and the optimization was performed through an in-house implementation of the simplex method (Nelder and Mead, 1965).

		Spanish	English
Train	Sentences	1.2M	
	Words	32M	31M
	Vocabulary	159K	111K
Dev	Sentences	1 122	699
	Words	26K	21K
Test	Sentences	1 117	894
	Words	26K	26K

Table 1: Statistics of the EPPS Corpora.

4 Structural comparison

Both approaches aim at improving accuracy by including word context in the model. However, the implementation of the models are quite different and may produce variations in several aspects.

Table 2 shows the effect on decoding time introduced through different settings of the beam size. Additionally, the number of available translation units is shown, corresponding to number of available phrases for the phrase-based system and 1gram, 2gram and 3gram entries for the N-gram-based system. Results are computed on the development set.

Task	Beam	Time(s)	Units
es→en	50	2,677	
	10	852	537k
	5	311	
en→es	50	2,689	
	10	903	594k
	5	329	
es→en	50	1,264	
	10	281	104k 288k 145k
	5	138	
en→es	50	1,508	
	10	302	118k 355k 178k
	5	155	

Table 2: Impact on efficiency of the beam size in PB (top) and NB system (bottom).

As it can be seen, the number of translation units is similar in both tasks for both systems ($537k \sim 537k$ for Spanish to English and $594k \sim 651k$ for English to Spanish) while the time consumed in decoding is clearly higher for the phrase-based system. This can be explained by the fact that in the phrase-based approach, the same translation can be hypothesized following several segmentations of the input sentence, as phrases appear (and are collected) from multiple segmentations of the training sentence pairs. In other words, the search graph seems to be overpopulated under the phrase-based approach.

Table 3 shows the effect on translation accuracy regarding the size of the beam in the search. Results are computed on the test set for the phrase-based and N-gram-based systems.

Results of the N-gram-based system show that decreasing the beam size produces a clear reduction of the accuracy results. The phrase-based system shows that accuracy results remain very similar under the different settings. The reason is found on how translation models are used in the search. In the phrase-based approach, every partial hypothesis

Task	Beam	BLEU	NIST	mWER
es→en	50	51.90	10.53	37.54
	10	51.93	10.54	37.49
	5	51.87	10.55	37.47
en→es	50	47.75	9.94	41.20
	10	47.77	9.96	41.09
	5	47.86	10.00	40.74
es→en	50	51.63	10.46	37.88
	10	51.50	10.45	37.83
	5	51.39	10.45	37.85
en→es	50	47.73	10.08	40.50
	10	46.82	9.97	41.04
	5	45.59	9.83	41.04

Table 3: Impact on accuracy of the beam size in PB (top) and NB system (bottom).

is scored uncontextualized, hence, a single score is used for a given partial hypothesis (phrase). In the N-gram-based approach, the model is intrinsically contextualized, which means that each partial hypothesis (tuple) depends on the preceding sequence of tuples. Thus, if a bad sequence of tuples (bad scored) is composed of a good initial sequence (well scored), it is placed on top of the first stacks (beam) and may cause the pruning of the rest of hypotheses.

5 Error analysis

In order to better asses the quality and the differences between the two systems, a human error analysis was carried out. The guidelines for this error analysis can be found in (Vilar et al., 2006). We randomly selected 100 sentences, which were evaluated by bilingual judges.

This analysis reveals that both systems produce the same kind of errors in general. However some differences were identified. For the English to Spanish direction the greatest problem is the correct generation of the right tense for verbs, with around 20% of all translation errors being of this kind. Reordering also poses an important problem for both phrase and N-gram-based systems, with 18% or 15% (respectively) of the errors falling into this category. Missing words is also an important problem. However, most of them (approximately two thirds for both systems) are filler words (i.e. words which do not convey meaning), that is, the meaning of the sentence is preserved. The most remarkable difference when comparing both systems is that the N-gram based system produces a relatively large amount of extra words (approximately 10%), while for the phrase-based system, this is only a minor problem (2% of the errors). In contrast the phrase-based system has more problems with incorrect translations, that is words for which a human can find a correspondence in the source text, but the translation is incorrect.

Similar conclusions can be drawn for the inverse direction. The verb generating problem is not so acute in this translation direction due to the much simplified morphology of English. An important problem is the generation of the right preposition.

The N-gram based system seems to be able to produce more accurate translations (reflected by a lower percentage of translation errors). However, it generates too many additional (and incorrect words) in the process. The phrase-based system, in contrast, counteracts this effect by producing a more direct correspondence with the words present in the source sentence at the cost of sometimes not being able to find the exact translation.

6 System Rescoring and Combination

Integration of both output translations in the search procedure is a complex task. Translation units of both models are quite different and generation histories pose severe implementation difficulties. We propose a method for combining the two systems at the level of N-best lists.

Some features that are useful for SMT are too complex for including them directly in the search process. A clear example are the features that require the entire target sentence to be evaluated, as this is not compatible with the pruning and recombination procedures that are necessary for keeping the target sentence generation process manageable. A possible solution for this problem is to apply sentence level re-ranking by using N-best lists.

6.1 Rescoring Criteria

The aim of the rescoring procedure is to choose the best translation candidate out of a given set of N possible translations. In our approach this translation candidates are produced independently by both of the systems and then combined by a simple concatenation[1]. In order for the hypothesis to have a comparable set of scores, we perform an additional "cross-rescoring" of the lists.

Given an N-best list of the phrase-based (N-gram-based) system, we compute the cost of each target sentence of this N-best list for the N-gram-based (phrase-based) system. However this computation is not possible in all cases. Table 4 shows the percentage of target sentences that the N-gram-based

[1]With removal of duplicates.

(phrase-based) system is able to produce given an N-best list of target sentences computed by the phrase-based (N-gram-based) system. This percentage is calculated on the development set.

The vocabulary of phrases is bigger than the vocabulary of tuples, due to the fact that phrases are extracted from multiple segmentations of the training sentence pairs. Hence, the number of sentences reproduced by the N-gram-based system is smaller than the number of sentences reproduced by the phrase-based system. Whenever a sentence can not be reproduced by a given system, the cost of the worst sentence in the N-best list is assigned to it.

Task	N-best	% NB	% PB
es→en	1000	37.5	57.5
en→es	1000	37.2	48.6

Table 4: Sentences (%) produced by each system.

6.2 Results

Table 5 shows results of the rescoring and system combination experiments on the test set. The first two rows include results of systems non-rescored and PB (NB) rescored by NB (PB). The third row corresponds to the system combination. Here, PB (NB) rescored by NB (PB) are simply merged and ranked by rescored score.

System	N-best	BLEU	NIST	mWER
Spanish-to-English				
PB	1	51.90	10.54	37.50
PB	1000	52.55	10.61	37.12
NB	1	51.63	10.46	37.88
NB	1000	52.25	10.55	37.43
PB+NB	2	51.77	10.49	37.68
PB+NB	2000	52.31	10.56	37.32
English-to-Spanish				
PB	1	47.75	9.94	41.2
PB	1000	48.46	10.13	39.98
NB	1	47.73	10.09	40.50
NB	1000	48.33	10.15	40.13
PB+NB	2	48.26	10.05	40.61
PB+NB	2000	48.54	10.16	40.00

Table 5: Rescoring and system combination results.

7 Discussion

The structural comparison has shown on the one hand that the N-gram-based system outperforms the phrase-based in terms of search time efficiency by avoiding the overpopulation problem presented in the phrase-based approach. On the other hand the phrase-based system shows a better performance when decoding under a highly constrained search.

A detailed error analysis has also been carried out in order to better determine the differences in performance of both systems. The N-gram based system produced more accurate translations, but also a larger amount of extra (incorrect) words when compare to the phrase-based translation system.

In section 6 we have presented a system combination method using a rescoring feature for each SMT system, i.e. the N-gram-based feature for the phrase-based system and vice-versa. For both systems, considering the feature of the opposite system leads to an improvement of BLEU score.

References

M.R. Costa-jussà, J.M. Crego, A. de Gispert, P. Lambert, M. Khalilov J.A.R. Fonollosa, J.B. Mariño, and R. Banchs. 2006. Talp phrase-based statistical machine translation and talp system combination the iwslt 2006. *IWSLT06*.

J. M. Crego, M. R. Costa-jussà, J. Mariño, and J. A. Fonollosa. 2005a. N-gram-based versus phrase-based statistical machine translation. *IWSLT05*, October.

J.M. Crego, J. Mariño, and A. de Gispert. 2005b. An Ngram-based statistical machine translation decoder. *ICSLP05*, April.

J.B. Mariño, R.E. Banchs, J.M. Crego, A. de Gispert, P. Lambert, J.A.R. Fonollosa, and M.R. Costa-jussà. 2006. N-gram based machine translation. *Computational Linguistics*, 32(4):527–549.

E. Matusov, N. Ueffing, and H. Ney. 2006. Computing consensus translation from multiple machine translation systems using enhanced hypotheses alignment. *EACL06*, pages 33–40.

J.A. Nelder and R. Mead. 1965. A simplex method for function minimization. *The Computer Journal*, 7:308–313.

F.J. Och. 2003. Giza++ software. http://www-i6.informatik.rwth-aachen.de/~och/ software/giza++.html.

A. Stolcke. 2002. Srilm - an extensible language modeling toolkit. *Proc. of the 7th Int. Conf. on Spoken Language Processing, ICSLP'02*, September.

David Vilar, Jia Xu, Luis Fernando D'Haro, and Hermann Ney. 2006. Error Analysis of Machine Translation Output. In *LREC06*, pages 697–702, Genoa, Italy, May.

Richard Zens and Hermann Ney. 2004. Improvements in phrase-based statistical machine translation. In *HLT04*, pages 257–264, Boston, MA, May.

Stating with Certainty or Stating with Doubt: Intercoder Reliability Results for Manual Annotation of Epistemically Modalized Statements

Victoria L. Rubin
Faculty of Information and Media Studies
University of Western Ontario
London, Ontario, Canada N6A 5B7
vrubin@uwo.ca

Abstract

Texts exhibit subtle yet identifiable modality about writers' estimation of how true each statement is (e.g., *definitely true* or *somewhat true*). This study is an analysis of such explicit certainty and doubt markers in epistemically modalized statements for a written news discourse. The study systematically accounts for five levels of writer's certainty (ABSOLUTE, HIGH, MODERATE, LOW CERTAINTY and UNCERTAINTY) in three news pragmatic contexts: perspective, focus, and time. The study concludes that independent coders' perceptions of the boundaries between shades of certainty in epistemically modalized statements are highly subjective and present difficulties for manual annotation and consequent automation for opinion extraction and sentiment analysis. While stricter annotation instructions and longer coder training can improve intercoder agreement results, it is not entirely clear that a five-level distinction of certainty is preferable to a simplistic distinction between statements with certainty and statements with doubt.

1 Introduction

1.1 Epistemic Modality, or Certainty

Text conveys more than just a writer's propositional context of assertions (Coates, 1987), e.g., *X is true*. Text can also transfer the writers' attitudes to the propositions, assessments of possibilities, and the writer's certainty, or lack thereof, in the validity of the truth of the statements, e.g., *X must be true*, *Y thinks that X is true*, or *perhaps X is true*. A statement is qualified in such a way (beyond its mere referential function) is modal, or epistemically modalized (Coates, 1987; Westney, 1986).

CERTAINTY, or EPISTEMIC MODALITY, concerns a linguistic expression of an estimation of the likelihood that a certain state of affairs is, has been, or will be true (Nuyts, 2001). Pragmatic and discourse literatures are abundant in discussions of epistemic modality (Coates, 1987; Nuyts, 2001); mood (Palmer, 1986); evidentiality and evidentials (Mushin, 2001); expressions of doubt and certainty (Holmes, 1982; Hoye, 1997) and hedging (Lackoff, 1972) and hedging in news writing (Hyland, 1999; Zuck & Zuck, 1986). Little attempt, however, has been made in natural language computing literature to manually annotate and consequently automate identification of statements with an explicitly expressed certainty or doubt, or shades of epistemic qualifications in between. This lack is possibly due to the complexity of computing epistemic interpretations in different pragmatic contexts; and due to unreliability of variety of linguistic expressions in English that could explicitly qualify a statement. Another complication is a lack of agreed-upon and easily identifiable discrete categories on the continuum from certainty to doubt. Several annotation projects have successfully addressed closely related subjective issues such as private states in news writing (Wiebe, Wilson, & Cardie, 2005) and hedging in scientific writing (Light, Qiu, & Srinivasan, 2004; Mercer, DiMarco, & Kroon, 2004). Having access to the opinion holder's evaluation of how true a statement is valuable in predicting reliability of arguments and claims, and stands to benefit the tasks of

Proceedings of NAACL HLT 2007, Companion Volume, pages 141–144,
Rochester, NY, April 2007.

opinion and sentiment analysis and extraction in natural language computing.

1.2 Certainty Level Scales

While there is an on-going discussion in pragmatic literature on whether epistemic modality markers should be arranged on a continuum or in discrete categories, there seems to be an agreement that there are at least three articulated points on a presumed continuum from certainty to doubt. Hoye (1997) suggested an epistemic trichotomy of CERTAINTY, PROBABILITY, and POSSIBILITY, consistent with Holmes' (1982) scale of certainty of assertions and negations where the writer asserts WITH CERTAINTY that a proposition is true or not true; or that the proposition is PROBABLY or POSSIBLY true or not true. In attitude and affect computational analysis literature, the context of extracting opinions from news article corpora, Rubin and colleagues (2004; 2005) extended Hoye-Holmes models by adding two extremes on the epistemic continuum scales: ABSOLUTE CERTAINTY (defined as a stated unambiguous indisputable conviction or reassurance) and UNCERTAINTY (defined as hesitancy or stated lack of clarity or knowledge), and re-defined the middle categories as HIGH CERTAINTY (i.e., high probability or firm knowledge), MODERATE CERTAINTY (i.e., estimation of an average likelihood or reasonable chances), and LOW CERTAINTY (i.e., distant possibility, see Fig. 1).

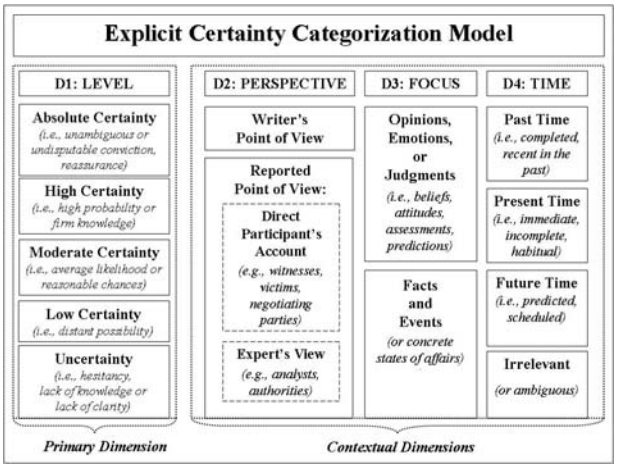


Figure 1. Revised Explicit Certainty Categorization Model (redrawn from Rubin, 2006).

While Rubin's (2006) model is primarily concerned with identification of certainty levels encoded in explicit certainty markers in propositions, it also takes into account three contextual dimensions relevant to news discourse. Perspective attributes explicit certainty either to the writer or two types of reported sources – direct participants and experts in a field. Focus separates certainty in facts and opinions. Time is an organizing principle of news production and presentation, and if relevant, is separated into past, present, or future.

2 Methodology

This study uses the above-described conceptual certainty categorization model to annotate a news dataset, and produce a typology of syntactic, semantic and lexical classes of certainty markers that map statements into 5 levels of certainty ranging from absolutely certain to uncertain.

The dataset consisted of 80 randomly selected articles (from the AQUAINT Corpus of English Texts, distributed by The New York Times Services in 2000). It constituted a total of 2,243 sentences, with 866 sentences in the editorials and 1377 sentence in the news reports (Rubin, 2006). A subset of 10 articles (272 sentences, about 12% of the full dataset) was analyzed by 4 independently trained annotators (excluding the author). The agreement results were evaluated in 2 consecutive intercoder reliability experiments.

2.1 Annotation Process

The manual annotation scheme was defined in the codebook instructions that specified the procedures for determining certainty-qualified statements, the order of assigning categories, and exemplified each certainty category (Rubin, 2006). In Experiment 1, three coders received individual one-hour training regarding the use of the annotation scheme, and were instructed to use the original codebook written in a general suggestive tone. In Experiment 2, the fourth annotator went through a more thorough five-hour training and used a revised, more rigidly-specified codebook with an alphabetized key-word index mapped certainty markers into 5 levels.

Each statement in a news article (be it a sentence or its constituent part such as a clause) was a potential locus of explicit certainty. In both experiments coders were asked to decide if a sentence had an explicit indication of a certainty level. If so, they then looked for explicit certainty markers that contributed to that indication. If a sentence contained a certainty marker, the annotators were in-

structed to consider such a sentence certainty-qualified. The statement was assigned a certainty level and placed in its pragmatic context (i.e., into one of the categories) within the perspective, focus, and time dimensions (see D2 – D4, Fig. 1) relevant to the news discourse. Each marker was only assigned one category from each dimension.

2.2 Intercoder Agreement Measures.

Each pair of coders were evaluated on whether they agreed regarding 1) the sentences that contained explicit certainty markers; 2) the specific certainty markers within agreed upon certainty-qualified sentences; and 3) classification of the agreed upon markers into one of the categories within each dimension (i.e., level, perspective, focus and time). The sentence and marker agreement measures were calculated with percent agreement. Partial word string matches were considered a marker match but were weight-adjusted. The agreed-upon marker category assignments were assessed in each pair of independent coders with Cohen's kappa statistic (Cohen, 1960), averaged, and compared to the author's annotation.

3 Results and Discussion

3.1 Typology of Certainty Markers

The content analysis of the dataset generated a group of 1,330 explicitly certainty-qualified sentences with 1,727 occurrences of markers. The markers were grouped into a typology of 43 syntactico-lexical classes; each class is likely to occur within one of the 5 levels of certainty. The typology will become a basis for an automated certainty identification algorithm. Among the most frequently used certainty markers are central modal auxiliary verbs (e.g., *must, could*), gradable adjectives in their superlative degree, and adverbial intensifiers (e.g., *much* and *so*), while adjectival downtoners (e.g., *feeble* + NP) and adverbial value disjuncts (e.g., *annoyingly*, *rightly*) are rarely used to express explicit certainty.

3.2 Intercoder Reliability Test Results

In Experiment 1, 1) three coders agreed on whether a sentences was modalized by an explicit certainty marker or not 71% of the time with 0.33 Cohen's kappa, on average. 2) Within agreed-upon certainly-qualified sentences, three coders agreed on actual certainty markers 54% of the time, on average, based on a combined count of the full and weight-adjusted partial matches. 3) In the categorization task for the agreed-upon markers, the three coders, on average, were able to reach a slight agreement in the level and focus dimensions (0.15 and 0.13 kappa statistics, respectively), and a fair agreement in perspective and time dimensions (0.44 and 0.41 kappa) according to the Landis and Koch (1977) agreement interpretation scale.

The subsequent Experiment 2 showed promising results in agreement on explicit certainty markers (67%) and overall ability to distinguish certainty-qualified statements from unmarked statements (0.51 kappa), and in the relatively intuitive categorization of the perspective dimension (0.65 kappa).

Although stricter instructions may have imposed a more orderly way of looking at the epistemic continuum, the 5 level certainty boundaries are still subject to individual perceptions (0.41 kappa) and may present difficulties in automation. In spite of its large inventory of certainty markers, English may not be precise enough to reliably distinguish multiple epistemic shades between certainty and doubt. Alternatively, people might be using same expressions but underlying categorization systems for different individuals do not overlap accurately. Recent pragmatic, discourse, and philosophy of language studies in mood and modality call for more comprehensive and truer to natural language description of epistemic modality in English reference grammar materials (Hoye, 2005). The latest modality scholarship will undoubtedly contribute to natural language applications such as opinion extraction and sentiment analysis.

Time categorization in the context of certainty remained a challenge in spite of more vigorous training in Experiment 2 (0.31 kappa). The interpretation of the reference point of "the present" in the reported speech and nested events can be ambiguous in the certainty identification task. Distinguishing facts versus opinions in combination with certainty identification also presented a particularly puzzling cognitive task (0.16 kappa), possibly due to necessity to evaluate closely related facets of a statement: whether the statement is purely factual, and how sure the author is about the proposition. The possibility of epistemically modalized facts is particularly intriguing.

4 Conclusions and Applications

This study reported the results of the manual annotation of texts in written news discourse, and identified the most prominent patterns and regularities in explicitly stated markers occurrences in modalized statements. The linguistic means of expressing varying levels of certainty are documented and arranged into the typology of syntactico-semantic classes. This study implies that boundaries between shades of certainty in epistemically modalized statements (such as probability and possibility) are highly subjective and present difficulties in manual annotation. This conclusion may warrant a simplification of the existing 5 certainty levels to a basic binary distinction between certainty and doubt. A baseline for future attempts to improve the calibration of levels and their boundaries was established. These modest intercoder reliability results attest to the complexity of the automation of the epistemically modalized statements ranging from certainty to doubt.

In the future studies, I intend to revise the number of the discrete categories on the epistemic continuum and further re-define certainty levels conceptually. I plan to further validate the collection of agreed-upon certainty markers on a much larger dataset and by using the typology as input data to machine learning algorithms for certainty identification and extraction.

References

Coates, J. (1987). Epistemic Modality and Spoken Discourse. *Transactions of the Philological Society*, 110-131.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement, 20*, 37-46.

Holmes, J. (1982). Expressing Certainty and Doubt in English. *RELC Journal, 13*(2), 9-29.

Hoye, L. (1997). *Adverbs and Modality in English*. London, New York: Longman.

Hoye, L. (2005). "You may think that; I couldn't possibly comment!" Modality Studies: Contemporary Research and Future Directions. Part II. *Journal of Pragmatics, 37*, 1481-1506.

Hyland, K. (1999). Academic attribution: Citation and the construction of disciplinary knowledge. *Applied Linguistics, 20*(3), 341-367.

Lackoff, G. (1972). *Hedges: a study of meaning criteria and the logic of fuzzy concepts.* Paper presented at the Chicago Linguistic Society Papers.

Landis, J., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics, 33*, 159-174.

Light, M., Qiu, X. Y., & Srinivasan, P. (2004). *The Language of Bioscience: Facts, Speculations, and Statements in Between.* Paper presented at the BioLINK 2004: Linking Biological Literature, Ontologies, and Databases.

Mercer, R. E., DiMarco, C., & Kroon, F. W. (2004). *The Frequency of Hedging Cues in Citation Contexts in Scientific Writing.* Paper presented at the Proceedings of the 17th Conference of the CSCSI/SCEIO (AI'2004).

Mushin, I. (2001). *Evidentiality and Epistemological Stance: Narrative Retelling* (Vol. 87). Amsterdam, Philadelphia: John Benjamins Publishing Company.

Nuyts, J. (2001). *Epistemic Modality, Language, and Conceptualization: A cognitive-pragmatic prespective* (Vol. 5). Amsterdam, Philadelphia: John Benjamin Publishing Company.

Palmer, F. R. (1986). *Mood and Modality*. Cambridge: Cambridge University Press.

Rubin, V. L. (2006). *Identifying Certainty in Texts.* Unpublished Doctoral Thesis, Syracuse University, Syracuse, NY.

Rubin, V. L., Kando, N., & Liddy, E. D. (2004). *Certainty Categorization Model.* Paper presented at the AAAI Spring Symposium: Exploring Attitude and Affect in Text: Theories and Applications, Stanford, CA.

Rubin, V. L., Liddy, E. D., & Kando, N. (2005). Certainty Identification in Texts: Categorization Model and Manual Tagging Results. In J. Wiebe (Ed.), *Computing Attitude and Affect in Text: Theory and Applications (The Information Retrieval Series)*: Springer-Verlag New York, Inc.

Westney, P. (1986). How to Be More-or-Less Certain in English - Scalarity in Epistemic Modality. *IRAL: International Review of Applied Linguistics in Language Teaching, 24*(4), 311-320.

Wiebe, J., Wilson, T., & Cardie, C. (2005). *Annotating Expressions of Opinions and Emotions in Language*. Netherlands: Kluwer Academic Publishers.

Zuck, J. G., & Zuck, L. V. (1986). *Hedging in Newswriting. beads or braclets? How do we approach LSP?* Paper presented at the Fifth European Symposium on LSP.

Joint Morphological-Lexical Language Modeling for Machine Translation

Ruhi Sarikaya **Yonggang Deng**

IBM T.J. Watson Research Center

Yorktown Heights, NY 10598

sarikaya@us.ibm.com ydeng@us.ibm.com

Abstract

We present a joint morphological-lexical language model (JMLLM) for use in statistical machine translation (SMT) of language pairs where one or both of the languages are morphologically rich. The proposed JMLLM takes advantage of the rich morphology to reduce the Out-Of-Vocabulary (OOV) rate, while keeping the predictive power of the whole words. It also allows incorporation of additional available semantic, syntactic and linguistic information about the morphemes and words into the language model. Preliminary experiments with an English to Dialectal-Arabic SMT system demonstrate improved translation performance over trigram based baseline language model.

1 Introduction

Statistical machine translation (SMT) methods have evolved from using the simple word based models (Brown et al., 1993) to phrase based models (Marcu and Wong, 2002; Koehn et al., 2004; Och and Ney, 2004). More recently, there is a significant effort focusing on integrating richer knowledge, such as syntactic parse trees (Huang and Knight, 2006) within the translation process to overcome the limitations of the phrase based models. The SMT has been formulated as a noisy channel model in which the target language sentence, e is seen as distorted by the channel into the foreign language f:

$$\hat{e} = \arg\max_{e} P(e \mid f) = \arg\max_{e} P(f \mid e)P(e)$$

where $P(f \mid e)$ is the translation model and $P(e)$ is language model of the target language. The overwhelming proportion of the SMT research has been focusing on improving the translation model. Despite several new studies (Kirchhoff and Yang, 2004; Schwenk et al., 2006), language modeling for SMT has not been receiving much attention. Currently, the state-of-the-art SMT systems have been using the standard word n-gram models. Since n-gram models learn from given data, a severe drop in performance may be observed if the target domain is not adequately covered in the training data. The coverage problem is aggravated for morphologically rich languages. Arabic is such a language where affixes are appended to the beginning or end of a stem to generate new words that indicate case, gender, tense etc. associated with the stem. Hence, it is natural that this leads to rapid vocabulary growth, which is accompanied by worse language model probability estimation due to data sparsity and high Out-Of-Vocabulary (OOV) rate.

Due to rich morphology, one would suspect that words may not be the best lexical units for Arabic, and perhaps morphological units would be a better choice. Recently, there have been a number of new methods using the morphological units to represent lexical items (Ghaoui et al., 2005; Xiang et al., 2006; Choueiter et al., 2006). Factored Language Models (FLMs) (Kirchhoff and Yang, 2004) share the same idea to some extent but here words are decomposed into a number of features and the resulting representation is used in a generalized back-off scheme to improve the robustness of probability estimates for rarely observed word *n*-grams.

In this study we propose a tree structure called Morphological-Lexical Parse Tree (MLPT) to combine the information provided by a morphological analyzer with the lexical information within a single Joint Morphological-Lexical Language Model (JMLLM). The MLPT allows us to include available syntactic and semantic information about the morphological segments[1] (i.e. prefix/stem/suffix), words or group of words. The JMLLM can also be used to guide the recognition for selecting high probability morphological sentence segmentations.

The rest of the paper is organized as follows. Section 2 provides a description of the morphological segmentation method. A short overview of Maximum Entropy modeling is given in Section 3. The proposed JMLLM is presented in Section 4. Section 5 introduces the SMT system and Section 6 describes the experimental results followed by the conclusions in Section 7.

2 Morphological Segmentation

Applying the morphological segmentation to data improves the coverage and reduces the OOV rate. In

[1] We use "Morphological Segment" and "Morpheme" interchangeably.

Proceedings of NAACL HLT 2007, Companion Volume, pages 145–148,
Rochester, NY, April 2007.

this study we use a rule-based morphological segmentation algorithm for Iraqi-Arabic (Afify et. al., 2006). This algorithm analyzes a given surface word, and generates one of the four possible segmentations: {*stem, prefix+stem, suffix+stem, prefix+stem+suffix*}. Here, *stem* includes those words that do not have any affixes. We use the longest prefixes (suffixes). Using finer affixes reduces the n-gram language model span, and leads to poor performance for a fixed n-gram size. Therefore, we predefine a set of prefixes and suffixes and perform blind word segmentation. In order to minimize the illegitimate segmentations we employ the following algorithm. Using the given set of prefixes and suffixes, a word is first blindly chopped to one of the four segmentations mentioned above. This segmentation is accepted if the following three rules apply:

(1) The resulting stem has more than two characters.
(2) The resulting stem is accepted by the Buckwalter morphological analyzer (Buckwalter, 2002).
(3) The resulting stem exists in the original dictionary.

The first rule eliminates many of the illegitimate segmentations. The second rule ensures that the word is a valid Arabic stem, given that the Buckwalter morphological analyzer covers all words in the Arabic language. Unfortunately, the fact that the stem is a valid Arabic stem does not always imply that the segmentation is valid. The third rule, while still not offering such guarantee, simply prefers keeping the word intact if its stem does not occur in the lexicon. In our implementation we used the following set of prefixes and suffixes for dialectal Iraqi:

- Prefix list: {chAl, bhAl, lhAl, whAl, wbAl, wAl, bAl, hAl, EAl, fAl, Al, cd, ll, b, f, c, d, w}.
- Suffix list: {thmA, tynA, hmA, thA, thm, tkm, tnA, tny,whA, whm, wkm, wnA, wny, An, hA, hm, hn, km, kn, nA, ny, tm, wA, wh, wk, wn, yn, tk, th, h, k, t, y}.

These affixes are selected based on our knowledge of their adequacy for dialectal Iraqi Arabic. In addition, we found in preliminary experiments that keeping the top-N frequent decomposable words intact led to better performance. A value of N=5000 was experimentally found to work well in practice. Using this segmentation method will produce prefixes and suffixes on the SMT output that are glued to the following or previous word to form meaningful words.

3 Maximum Entropy Modeling

The Maximum Entropy (MaxEnt) method is an effective method to combine multiple information sources (features) in statistical modeling and has been used widely in many areas of natural language processing (Berger et al.,, 2000). The MaxEnt modeling produces a probability model that is as uniform as possible while matching empirical feature expectations exactly:

$$P(o \mid h) = \frac{e^{\sum_i \lambda_i f_i(o,h)}}{\sum_{o'} e^{\sum_j \lambda_i f_j(o',h)}}$$

which describes the probability of a particular outcome (e.g. one of the morphemes) given the history (*h*) or context. Notice that the denominator includes a sum over all possible outcomes, *o'*, which is essentially a normalization factor for probabilities to sum to 1. The indicator functions f_i or features are "activated" when certain outcomes are generated for certain context. The MaxEnt model is trained using the Improved Iterative Scaling algorithm.

4 Joint Morphological-Lexical Language Modeling

The purpose of morphological analysis is to split a word into its constituting segments. Hence, a set of segments can form a meaningful lexical unit such as a word. There may be additional information for words or group of words, such as part-of-speech (POS) tags, syntactic (from parse tree) and semantic information, or morpheme and word attributes. For example, in Arabic and to a certain extent in French, some words can be masculine/feminine or singular/plural. All of these information sources can be represented using a -what we call- Morphological-Lexical Parse Tree (MLPT). MLPT is a tree structured joint representation of lexical, morphological, attribute, syntactic and semantic content of the sentence. An example of a MLPT for an Arabic sentence is shown in Fig. 1. The leaves of the tree are morphemes that are predicted by the language model. Each morphological segment has one of the three attributes: {*prefix, stem, suffix*} as generated by the morphological analysis mentioned in Sec. 2. Each word can take three sets of attributes: *{type, gender, number}.* Word *type* can be considered as POS, but here we consider only nouns (N), verbs (V) and the rest are labeled as "other" (O). *Gender* can be masculine (M) or feminine (F). *Number* can be singular (S), plural (P) or double (D) (this is specific to Arabic). For example, NMP label for the first[2] word, شباب, shows that this word is a noun (N), male (M), plural (P). Using the information represented in MLPT for Arabic language modeling provides a back-off for smooth probability estimation even for those words that are not seen before.

The JMLLM integrates the local morpheme and word *n*-grams, morphological dependencies and attribute information associated with morphological segments and words, which are all represented in the MLPT using the MaxEnt framework. We trained JMLLM for Iraqi-

[2] In Arabic text is written (read) from right-to-left.

Arabic speech recognition task (Sarikaya et al., 2007), and obtained significant improvements over word and morpheme based trigram language models.

We can construct a single probability model that models the joint probability of all of the available information sources in the MLPT. To compute the joint probability of the morpheme sequence and its MLPT, we use features extracted from MLPT. Even though the framework is generic to jointly represent the information sources in the MLPT, in this study we limit ourselves to using only lexical and morphological content of the sentence, along with the morphological attributes simply because the lexical attributes are not available yet and we are in the process of labeling them. Therefore, the information we used from MLPT in Fig. 1 uses everything but the second row that contains lexical attributes (NFS, VFP, NFS, and NMP).

Using the morphological segmentation improves the coverage, for example, splitting the word, بالقهوة as بال (prefix) and قهوة (stem) as in Fig. 1, allows us to decode other combinations of this stem with the prefix and suffix list provided in Sec.2. These additional combinations certainly cover those words that are not seen in the unsegmented training data.

The first step in building the MaxEnt model is to represent a MLPT as a sequence of morphological segments, morphological attributes, words, and word attributes using a bracket notation. Converting the MLPT into a text sequence allows us to group the semantically related morphological segments and their attributes. In this notation, each morphological segment is associated (this association is denoted by "=") with an attribute (i.e. prefix/stem/suffix) and the lexical items are represented by opening and closing tokens, [WORD and WORD] respectively. The parse tree given in Fig. 1 can be converted into a token sequence in text format as follows:

[!S! [NMP شباب=stem NMP] [NFS [المنطقة ال=prefix منطقة=stem المنطقة] NFS] [VFP [يقعدون يقعدون=suffix ون=stem قعد=prefix ي] VFP] [NFS [بالقهوة بال=prefix بالقهوة=stem بالقهوة] NFS] !S!]

This representation uniquely defines the MLPT given in Fig. 1. Given the bracket notation of the text, JMLLM can be trained in two ways with varying degrees of "*tightness of integration*". A relatively "*loose integration"* involves using only the leaves of the MLPT as the model output and estimating *P(M|MLPT),* where *M* is the morpheme sequence. In this case JMLLM predicts only morphemes. A *tight integration* method would require every token in the bracket representation to be an outcome of the joint model. In our preliminary experiments we chose the *loose integration* method, simply because the model training time was significantly faster than that for the *tight integration*. segment. The JMLLM can employ any type of questions one can derive from MLPT for predicting the next morphological segment. In addition to regular trigram questions about previous morphological segments, questions about the attributes of the previous morpho-

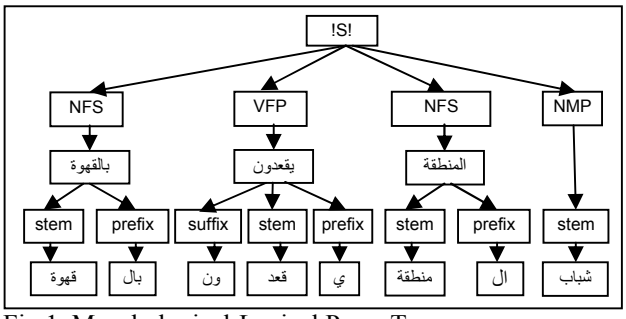


Fig 1. Morphological-Lexical Parse Tree.

logical segments, parent lexical item and attributes of the parent lexical item can be used. Obviously joint questions combining these information sources are also used. Obviously joint questions combining these information sources are also used. These questions include (1) previous morpheme m_{i-1} and current active parent word (w_i) (2) m_{i-1}, w_i and previous morpheme attribute (ma_{i-1}). (3) ma_{i-1}, ma_{i-2}, w_i ,*lexical attribute (* wa_i *) and* m_{i-1}, m_{i-2} .

The history given in $P(o \mid h)$ consists of answers to these questions. In our experiments, we have not exhaustively searched for the best feature set but rather used a small subset of these features which we believed to be helpful in predicting the next morpheme. The language model score for a given morpheme using JMLLM is conditioned not only on the previous morphemes but also on their attributes, and the lexical items and their attributes. As such, the language model scores are smoother compared to *n*-gram models especially for unseen lexical items.

5 Statistical Machine Translation System

Starting from a collection of parallel sentences, we trained word alignment models in two translation directions, from English to Iraqi Arabic and from Iraqi Arabic to English, and derived two sets of Viterbi alignments. By combining word alignments in two directions using heuristics (Och and Ney, 2003), a single set of static word alignments was then formed. All phrase pairs which respect to the word alignment boundary constraint were identified and pooled together to build phrase translation tables with the Maximum Likelihood criterion. The maximum number of words in Arabic phrases was set to 5.

Our decoder is the phrase-based multi-stack implementation of log-linear models similar to Pharaoh (Koehn et al, 2004). Like most other MaxEnt-based decoders, active features in our decoder include translation models in two directions, lexicon weights in two

directions, language model, distortion model, and sentence length penalty.

6 Experiments

The parallel corpus has 459K utterance pairs with 90K words (50K morphemes). The Iraqi-Arabic language model training data is slightly larger than the Iraqi-Arabic side of the parallel corpus and it has 2.8M words with 98K unique lexical items. The morphologically analyzed training data has 2.6M words with 58K unique vocabulary items. A statistical trigram language model using Modified Knesser-Ney smoothing has been built for the morphologically segmented data. The test data consists of 2242 utterances (3474 unique words). The OOV rate for the unsegmented test data is 8.7%, the corresponding number for the morphologically analyzed data is 7.4%. Hence, morphological segmentation reduces the OOV rate by 1.3% (15% relative), which is not as large reduction as compared to training data (about 40% relative reduction). We believe this would limit the potential improvement we could get from JMLLM, since JMLLM is expected to be more effective compared to word n-gram models, when the OOV rate is significantly reduced after segmentation.

We measure translation performance by the BLEU score (Papineni *et al*, 2002) with one reference for each hypothesis. In order to evaluate the performance of the JMLLM, a translation N-best list (N=10) is generated using the baseline Morpheme-trigram language model. First, on a heldout development data all feature weights including the language model weight are optimized to maximize the BLEU score using the downhill simplex method (Och and Hey, 2002). These weights are fixed when the language models are used on the test data. The translation BLEU (%) scores are given in Table 1. The first entry (37.59) is the oracle BLEU score for the N-best list. The baseline morpheme-trigram achieved 29.63, word-trigram rescoring improved the BLEU score to 29.91. The JMLLM achieved 30.20 and log-linear interpolation with the morpheme-trigram improved the BLEU score to 30.41.

Table 1. SMT N-best list rescoring.

LANGUAGE MODELS	BLEU (%)
N-best Oracle	37.59
Morpheme-trigram	29.63
Word-trigram	29.91
JMLLM	**30.20**
JMLLM + Morpheme-Trigram	**30.41**

7 Conclusions

We presented a new language modeling technique called Joint Morphological-Lexical Language Modeling (JMLLM) for use in SMT. JMLLM allows joint modeling of lexical, morphological and additional information sources about morphological segments, lexical items and sentence. The translation results demonstrate that joint modeling provides encouraging improvement over morpheme based language model. Our future work will be directed towards tight integration of all available information by predicting the entire MLPT (besides leaves).

References

P. Brown er al.,. 1993. The mathematics of statistical machine translation. *Computational Linguistics*, 19(2):263–311.

A. Berger, S. Della Pietra and V. Della Pietra, "A Maximum Entropy Approach to Natural Language Processing," Computational Linguistics, vol. 22, no. 1, March 1996

T. Buckwalter. 2002. Buckwalter Arabic morphological analyzer version 1.0, *LDC2002L49 and ISBN 1-58563-257-0*, 2002.

G. Choueiter, D. Povey, S.F. Chen, and G. Zweig, 2006. Morpheme-based language modeling for Arabic LVCSR. *ICASSP'06*, Toulouse, France, 2006.

A. Ghaoui, F. Yvon, C. Mokbel, and G. Chollet, 2005. On the use of morphological constraints in N-gram statistical language model, *Eurospeech'05*, Lisbon, Portugal, 2005.

B. Huang and K. Knight. 2006. Relabeling Syntax Trees to Improve Syntax-Based Machine Translation Quality. In HLT/NAACL.

B. Xiang, K. Nguyen, L. Nguyen, R. Schwartz, J. Makhoul, 2006. Morphological decomposition for Arabic broadcast news transcription", *ICASSP'06*, Toulouse, France, 2006.

K. Kirchhoff and M. Yang. 2005. Improved language modeling for statistical machine translation. In *ACL'05 workshop on Building and Using Parallel Text*, pages 125–128.

P. Koehn, F. J. Och, and D. Marcu. 2004. Pharaoh: A beam search decoder for phrase based statistical machine translation models. In *Proc. of 6th Conf. of AMTA.*

F. J. Och and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *ACL*, pages 295–302, University of Pennsylvania.

F. J. Och and H. Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. Comp. Linguistics, 29(1):9--51.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. BLEU: a Method for Automatic Evaluation of machine translation. In ACL 02, pages 311–318

H. Schwenk, D. D´echelotte and J-L. Gauvain. 2006. Continuous space language models for statistical machine translation. In ACL/*COLING*, pages 723–730.

M. Afify, R. Sarikaya, H-K. J. Kuo, L. Besacier and Y. Gao. 2006. On the Use of Morphological Analysis for Dialectal Arabic Speech Recognition, In Interspeech-2006, Pittsburgh PA.

R. Sarikaya, M .Afify and Y. Gao. 2007. Joint Morphological-Lexical Modeling (JMLLM) for Arabic. ICASSP 2007, Honolulu Hawaii.

Agenda-Based User Simulation for Bootstrapping a POMDP Dialogue System

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye and Steve Young
Cambridge University Engineering Department
Trumpington Street, Cambridge, CB21PZ, United Kingdom
{js532, brmt2, kw278, hy216, sjy}@eng.cam.ac.uk

Abstract

This paper investigates the problem of bootstrapping a statistical dialogue manager without access to training data and proposes a new probabilistic agenda-based method for simulating user behaviour. In experiments with a statistical POMDP dialogue system, the simulator was realistic enough to successfully test the prototype system and train a dialogue policy. An extensive study with human subjects showed that the learned policy was highly competitive, with task completion rates above 90%.

1 Background and Introduction

1.1 Bootstrapping Statistical Dialogue Managers

One of the key advantages of a statistical approach to Dialogue Manager (DM) design is the ability to formalise design criteria as objective reward functions and to learn an optimal dialogue policy from real dialogue data. In cases where a system is designed from scratch, however, it is often the case that no suitable in-domain data is available for training the DM. Collecting dialogue data without a working prototype is problematic, leaving the developer with a classic chicken-and-egg problem.

Wizard-of-Oz (WoZ) experiments can be carried out to record dialogues, but they are often time-consuming and the recorded data may show characteristics of human-human conversation rather than typical human-computer dialogue. Alternatively, human-computer dialogues can be recorded with a handcrafted DM prototype but neither of these two methods enables the system designer to test the implementation of the statistical DM and the learning algorithm. Moreover, the size of the recorded corpus (typically $\ll 10^3$ dialogues) usually falls short of the requirements for training a statistical DM (typically $\gg 10^4$ dialogues).

1.2 User Simulation-Based Training

In recent years, a number of research groups have investigated the use of a two-stage simulation-based setup. A statistical user model is first trained on a limited amount of dialogue data and the model is then used to simulate dialogues with the interactively learning DM (see Schatzmann et al. (2006) for a literature review).

The simulation-based approach assumes the presence of a small corpus of suitably annotated in-domain dialogues or out-of-domain dialogues with a matching dialogue format (Lemon et al., 2006). In cases when no such data is available, handcrafted values can be assigned to the model parameters given that the model is sufficiently simple (Levin et al., 2000; Pietquin and Dutoit, 2005) but the performance of dialogue policies learned this way has not been evaluated using real users.

1.3 Paper Outline

This paper presents a new probabilistic method for simulating user behaviour based on a compact representation of the *user goal* and a stack-like *user agenda*. The model provides an elegant way of encoding the relevant dialogue history from a user's point of view and has a very small parameter set so that manually chosen priors can be used to bootstrap the DM training and testing process.

In experiments presented in this paper, the agenda-based simulator was used to train a statistical POMDP-based (Young, 2006; Young et al., 2007) DM. Even without any training of its model parameters, the agenda-based simulator was able to produce dialogue behaviour realistic enough to train a competitive dialogue policy. An extensive study[1] with 40 human subjects showed that task completion with the learned policy was above 90% despite a mix of native and non-native speakers.

[1]This research was partly funded by the EU FP6 TALK Project. The system evaluation was conducted in collaboration with O. Lemon, K. Georgila and J. Henderson at Edinburgh University and their work is gratefully acknowledged.

Proceedings of NAACL HLT 2007, Companion Volume, pages 149–152,
Rochester, NY, April 2007.

2 Agenda-Based Simulation

2.1 User Simulation at a Semantic Level

Human-machine dialogue can be formalised on a semantic level as a sequence of state transitions and dialogue acts[2]. At any time t, the user is in a state S, takes action a_u, transitions into the intermediate state S', receives machine action a_m, and transitions into the next state S'' where the cycle restarts.

$$S \rightarrow a_u \rightarrow S' \rightarrow a_m \rightarrow S'' \rightarrow \cdots \quad (1)$$

Assuming a Markovian state representation, user behaviour can be decomposed into three models: $P(a_u|S)$ for action selection, $P(S'|a_u, S)$ for the state transition into S', and $P(S''|a_m, S')$ for the transition into S''.

2.2 Goal- and Agenda-Based State Representation

Inspired by agenda-based methods to dialogue management (Wei and Rudnicky, 1999) the approach described here factors the user state into an agenda A and a goal G.

$$S = (A, G) \qquad \text{and} \qquad G = (C, R) \quad (2)$$

During the course of the dialogue, the goal G ensures that the user behaves in a consistent, goal-directed manner. G consists of constraints C which specify the required venue, eg. a centrally located bar serving beer, and requests R which specify the desired pieces of information, eg. the name, address and phone number (cf. Fig. 1).

The user agenda A is a stack-like structure containing the pending user dialogue acts that are needed to elicit the information specified in the goal. At the start of the dialogue a new goal is randomly generated using the system database and the agenda is initially populated by converting all goal constraints into *inform* acts and all goal requests into *request* acts. A *bye* act is added at the bottom of the agenda to close the dialogue.

As the dialogue progresses the agenda and goal are dynamically updated and acts are selected from the top of the agenda to form user acts a_u. In response to incoming machine acts a_m, new user acts are pushed onto the agenda and no longer relevant ones are removed. The agenda thus serves as a convenient way of tracking the progress of the dialogue as well as encoding the relevant dialogue history. Acts can also be temporarily stored when actions of higher priority need to be issued first, hence providing the simulator with a simple model of user memory[3].

[2]In this paper, the terms *dialogue act* and *dialogue action* are used interchangeably. The notation *act(a=x, b=y,...)* is used to represent a dialogue act of a given type *act* (such as *inform* or *request* with items $a = x$, $b = y$, etc.

[3]See Fig. 1, turns 1-3, for an example: System turn 1 "*What pricerange?*" triggers the user act *inform(pricerange=cheap)* to be pushed onto the agenda but it is not executed until turn 3 because *negate(drinks=beer)* is issued first.

$C_0 = \begin{bmatrix} type = bar \\ drinks = beer \\ area = central \end{bmatrix}$

$R_0 = \begin{bmatrix} name = \\ addr = \\ phone = \end{bmatrix}$

Sys 0: Hello, how may I help you?

$A_1 = \begin{bmatrix} inform(type = bar) \\ inform(drinks = beer) \\ inform(area = central) \\ request(name) \\ request(addr) \\ request(phone) \\ bye() \end{bmatrix}$

Usr 1: I'm looking for a nice bar serving beer.

Sys 1: Ok, a wine bar. What pricerange?

$A_2 = \begin{bmatrix} negate(drinks = beer) \\ inform(pricerange = cheap) \\ inform(area = central) \\ request(name) \\ request(addr) \\ request(phone) \\ bye() \end{bmatrix}$

Usr 2: No, beer please!

Sys 2: A bar serving beer, correct?

$A_3 = \begin{bmatrix} affirm() \\ inform(pricerange = cheap) \\ inform(area = central) \\ request(name) \\ request(addr) \\ request(phone) \\ bye() \end{bmatrix}$

Usr 3: Yeah something cheap in the town centre.

Sys 3: Murphy's on Main Square serves cheap beer.

$A_4 = \begin{bmatrix} request(phone) \\ bye() \end{bmatrix}$

Usr 4: Ok, and what's the phone number?

Sys 4: The number is 796 69 94.

$A_5 = \begin{bmatrix} bye() \end{bmatrix}$

Usr 5: Thanks, goodbye!

Figure 1: Sample dialogue and agenda sequence

2.3 User Act Selection

At any time during the dialogue, the updated agenda of length N contains all dialogue acts the user intends to convey to the system. Since the agenda is ordered according to priority, with $A[N]$ denoting the top and $A[1]$ denoting the bottom item, selecting the next user act simplifies to popping n items off the top of the stack. Hence, letting $a_u[i]$ denote the ith item in the user act a_u

$$a_u[i] := A[N - n + i] \quad \forall i \in [1..n], 1 \leq n \leq N. \quad (3)$$

Using $A[N–n+1..N]$ as a Matlab-like shorthand nota-

tion for the top n items on A, the action selection model becomes a Dirac delta function

$$P(a_u|S) = P(a_u|A, G) = \delta(a_u, A[N-n+1..N]) \quad (4)$$

where the random variable n corresponds to the level of initiative taken by the simulated user. In a statistical model the probability distribution over integer values for n should be conditioned on A and learned from dialogue data. For the purposes of bootstrapping the system, n can be assumed independent of A and any distribution $P(n)$ that places the majority of its probability mass on small values of n can be used.

2.4 State Transition Model

The factorisation of S into A and G can now be applied to the state transition models $P(S'|a_u, S)$ and $P(S''|a_m, S')$. Letting A' denote the agenda after selecting a_u (as explained in the previous subsection) and using $N' = N - n$ to denote the size of A', we have

$$A'[i] \quad := \quad A[i] \quad \forall i \in [1..N']. \quad (5)$$

Using this definition of A' and assuming that the goal remains constant when the user executes a_u, the first state transition depending on a_u simplifies to

$$\begin{aligned} P(S'|a_u, S) &= P(A', G'|a_u, A, G) \\ &= \delta(A', A[1..N'])\delta(G', G). \end{aligned} \quad (6)$$

Using $S = (A, G)$, the chain rule of probability, and reasonable conditional independence assumptions, the second state transition based on a_m can be decomposed into *goal update* and *agenda update* modules:

$$\begin{aligned} &P(S''|a_m, S') \\ &= \underbrace{P(A''|a_m, A', G'')}_{\text{agenda update}} \underbrace{P(G''|a_m, G')}_{\text{goal update}}. \end{aligned} \quad (7)$$

When no restrictions are placed on A'' and G'', the space of possible state transitions is vast. The model parameter set is too large to be handcrafted and even substantial amounts of training data would be insufficient to obtain reliable estimates. It can however be assumed that A'' is derived from A' and that G'' is derived from G' and that in each case the transition entails only a limited number of well-defined atomic operations.

2.5 Agenda Update Model for System Acts

The agenda transition from A' to A'' can be viewed as a sequence of push-operations in which dialogue acts are added to the top of the agenda. In a second "clean-up" step, duplicate dialogue acts, *null()* acts, and unnecessary *request()* acts for already filled goal request slots must be removed but this is a deterministic procedure so that it can be excluded in the following derivation for simplicity. Considering only the push-operations, the items 1 to N' at the bottom of the agenda remain fixed and the update model can be rewritten as follows:

$$\begin{aligned} &P(A''|a_m, A', G'') \\ &= P(A''[1..N'']|a_m, A'[1..N'], G'') \quad (8) \\ &= P(A''[N'+1..N'']|a_m, G'') \\ &\quad \cdot \ \delta(A''[1..N'], A'[1..N']). \quad (9) \end{aligned}$$

The first term on the RHS of Eq. 9 can now be further simplified by assuming that every dialogue act item in a_m triggers one push-operation. This assumption can be made without loss of generality, because it is possible to push a *null()* act (which is later removed) or to push an act with more than one item. The advantage of this assumption is that the known number M of items in a_m now determines the number of push-operations. Hence $N'' = N' + M$ and

$$\begin{aligned} &P(A''[N'+1..N'']|a_m, G'') \\ &= P(A''[N'+1..N'+M]|a_m[1..M], G'') \quad (10) \\ &= \prod_{i=1}^{M} P(A''[N'+i]|a_m[i], G'') \quad (11) \end{aligned}$$

The expression in Eq. 11 shows that each item $a_m[i]$ in the system act triggers one push operation, and that this operation is conditioned on the goal. This model is now simple enough to be handcrafted using heuristics. For example, the model parameters can be set so that when the item *x=y* in $a_m[i]$ violates the constraints in G'', one of the following is pushed onto A'': *negate()*, *inform(x=z)*, *deny(x=y, x=z)*, etc.

2.6 Goal Update Model for System Acts

The goal update model $P(G''|a_m, G')$ describes how the user constraints C' and requests R' change with a given machine action a_m. Assuming that R'' is conditionally independent of C' given C'' it can be shown that

$$\begin{aligned} &P(G''|a_m, G') \\ &= P(R''|a_m, R', C'')P(C''|a_m, R', C'). \end{aligned} \quad (12)$$

To restrict the space of transitions from R' to R'' it can be assumed that the request slots are independent and each slot (eg. *addr,phone,etc.*) is either filled using information in a_m or left unchanged. Using $R[k]$ to denote the k'th request slot, we approximate that the value of $R''[k]$ only depends on its value at the previous time step, the value provided by a_m, and $\mathcal{M}(a_m, C'')$ which indicates a match or mismatch between the information given in a_m and the goal constraints.

$$\begin{aligned} &P(R''|a_m, R', C'') \\ &= \textstyle\prod_k P(R''[k]|a_m, R'[k], \mathcal{M}(a_m, C'')). \end{aligned} \quad (13)$$

To simplify $P(C''|a_m, R', C')$ we assume that C'' is derived from C' by either adding a new constraint, setting an existing constraint slot to a different value (eg. *drinks=dontcare*), or by simply changing nothing. The choice of transition does not need to be conditioned on the full space of possible a_m, R' and C'. Instead it can be conditioned on simple boolean flags such as "Does a_m ask for a slot in the constraint set?", "Does a_m signal that no item in the database matches the given constraints?", etc. The model parameter set is then sufficiently small for handcrafted values to be assigned to the probabilities.

3 Evaluation

3.1 Training the HIS Dialogue Manager

The Hidden Information State (HIS) model is the first trainable and scalable implementation of a statistical spoken dialog system based on the Partially-Observable Markov Decision Process (POMDP) model of dialogue (Young, 2006; Young et al., 2007; Williams and Young, 2007). POMDPs extend the standard Markov-Decision-Process model by maintaining a *belief space*, i.e. a probability distribution over dialogue states, and hence provide an explicit model of the uncertainty present in human-machine communication.

The HIS model uses a grid-based discretisation of the continuous state space and online ϵ-greedy policy iteration. Fig. 2 shows a typical training run over 60,000 simulated dialogues, starting with a random policy. User goals are randomly generated and an (arbitrary) reward function assigning 20 points for successful completion and -1 for every dialogue turn is used. As can be seen, dialogue performance (defined as the average reward over 1000 dialogues) converges after roughly 25,000 iterations and asymptotes to a value of approx. 14 points.

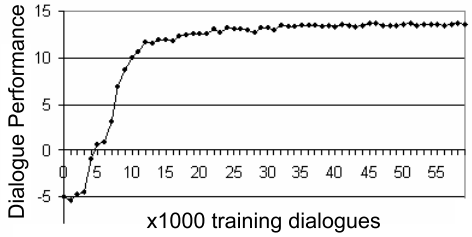


Figure 2: Training a POMDP system

3.2 Experimental Evaluation and Results

A prototype HIS dialogue system with a learned policy was built for the Tourist Information Domain and extensively evaluated with 40 human subjects including native and non-native speakers. A total of 160 dialogues with 21667 words was recorded and the average Word-Error-Rate was 29.8%. Task scenarios involved finding a specific bar, hotel or restaurant in a fictitious town (eg. the address of a cheap, Chinese restaurant in the west).

The performance of the system was measured based on the recommendation of a correct venue, i.e. a venue matching all constraints specified in the given task (all tasks were designed to have a solution). Based on this definition, 145 out of 160 dialogues (90.6%) were completed successfully, and the average number of turns to completion was 5.59 (if no correct venue was offered the full number of turns was counted).

4 Summary and Future Work

This paper has investigated a new agenda-based user simulation technique for bootstrapping a statistical dialogue manager without access to training data. Evaluation results show that, even with manually set model parameters, the simulator produces dialogue behaviour realistic enough for training and testing a prototype system. While the results demonstrate that the learned policy works well for real users, it is not necessarily optimal. The next step is hence to use the recorded data to train the simulator, and to then retrain the DM policy.

References

O. Lemon, K. Georgila, and J. Henderson. 2006. Evaluating Effectiveness and Portability of Reinforcement Learned Dialogue Strategies with real users: the TALK TownInfo Evaluation. In *Proc. of IEEE/ACL SLT*, Palm Beach, Aruba.

E. Levin, R. Pieraccini, and W. Eckert. 2000. A Stochastic Model of Human-Machine Interaction for Learning Dialog Strategies. *IEEE Trans. on Speech and Audio Processing*, 8(1):11–23.

O. Pietquin and T. Dutoit. 2005. A probabilistic framework for dialog simulation and optimal strategy learning. *IEEE Trans. on Speech and Audio Processing, Special Issue on Data Mining of Speech, Audio and Dialog*.

J. Schatzmann, K. Weilhammer, M.N. Stuttle, and S. Young. 2006. A Survey of Statistical User Simulation Techniques for Reinforcement-Learning of Dialogue Management Strategies. *Knowledge Engineering Review*, 21(2):97–126.

X Wei and AI Rudnicky. 1999. An agenda-based dialog management architecture for spoken language systems. In *Proc. of IEEE ASRU*. Seattle, WA.

J. Williams and S. Young. 2007. Partially Observable Markov Decision Processes for Spoken Dialog Systems. *Computer Speech and Language*, 21(2):231–422.

S. Young, J. Schatzmann, K. Weilhammer, and H. Ye. 2007. The Hidden Information State Approach to Dialog Management. In *Proc. of ICASSP*, Honolulu, Hawaii.

S. Young. 2006. Using POMDPs for Dialog Management. In *Proc. of IEEE/ACL SLT*, Palm Beach, Aruba.

Reversible Sound-to-letter/Letter-to-sound Modeling based on Syllable Structure *

Stephanie Seneff
Spoken Language Systems Group
MIT Computer Science and Artificial Intelligence Laboratory
The Stata Center, 32 Vassar Street, Cambridge, MA 02139
seneff@csail.mit.edu

Abstract

This paper describes a new grapheme-to-phoneme framework, based on a combination of formal linguistic and statistical methods. A context-free grammar is used to parse words into their underlying syllable structure, and a set of subword "spellneme" units encoding both phonemic and graphemic information can be automatically derived from the parsed words. A statistical n-gram model can then be trained on a large lexicon of words represented in terms of these linguistically motivated subword units. The framework has potential applications in modeling unknown words and in linking spoken spellings with spoken pronunciations for fully automatic new-word acquisition via dialogue interaction. Results are reported on sound-to-letter experiments for the nouns in the Phonebook corpus.

1 Introduction

Spoken dialogue systems are emerging as an effective means for humans to access information spaces through natural spoken interaction with computers. A significant enhancement to the usability of such systems would be the automatic acquisition of new knowledge through spoken interaction with its end users. Such knowledge would include both the spelling and pronunciation of a new word, ideally leading to a successful match to an entry in a large external database. To take advantage of an integrated approach to recognizing the spoken and spelled forms of a new word, there is a need for a high-quality reversible phoneme-grapheme mapping system. This is a difficult task for English due to the many inconsistencies in letter-to-sound rules as a consequence of borrowings from multiple language groups.

It is also increasingly the case that dialogue systems must dynamically adjust the recognizer vocabulary to handle changing database contents. If a system can reliably predict the pronunciation of a new word algorithmically, especially if substantiated by a spoken pronunciation of the word during active usage, it will be far more effective in satisfying changing user needs.

In this paper, we describe a new reversible grapheme-to-phoneme framework based on combining formal linguistic knowledge with statistical data-driven techniques. We first describe and motivate our choice for the linguistic model. Section 3 describes the iterative process for obtaining a subword baseforms lexicon used to train the statistical model. Sections 4 and 5 present experiments and results for sound-to-letter modeling on 5000 nouns. We conclude after a brief section on related work.

2 Linguistic Model

Our linguistic model is based on syllable structure, but we felt that whole-syllable units would be too large to adequately generalize to unseen data. We thus decided to decompose syllables into onsets and

*This research was supported by the Industrial Technology Research Institute (ITRI) in Taiwan.

Proceedings of NAACL HLT 2007, Companion Volume, pages 153–156,
Rochester, NY, April 2007.

rhyme1	onset	rhyme	usyl	rhyme	usyl	ambi	rhyme
-aek	s+	-ehl	-axr	-aam	-ax+	tf	-er+
a c	c	e l	e r	o m	e	t	e r

Figure 1: Linguistic representation for the word "accelerometer," illustrating the structure of our model.

rhymes, which would then become subword pronunciation units in a lexical baseforms file. These subword units would, in turn, be specified in terms of phonemic baseforms in a separate subword lexicon. Thus the words in our training set are represented in terms of subword units, which are converted into phonemic baseforms by simple lookup of the subword pronunciations.

A difficult aspect for English is to decide where to place the syllable boundary within a sequence of intersyllabic consonants. To guide this decision, we made use of sonority constraints combined with maximal stress and maximal onset principles. For a select subset of intersyllable consonants, we invoke the special category "ambi" for "ambisyllabic," to allow the consonant to be ambiguously assigned. In addition to onset and rhyme, we also include the category "affix," to account for those instances of (usually coronal) consonants that would lead to a violation of sonority principles in the coda position (e.g., "fif*ths*," "kep*t*", etc.), following linguistic theory (Blevins, 1995).

We decided to distinguish the first stressed and the first unstressed syllable from all other stressed and unstressed syllables in the word, in order to encode separate statistics for the privileged first position. We also combined onset and rhyme into a single whole syllable unit for a selected subset of relatively frequent unstressed syllables. In total, our current inventory consists of 678 unique symbols.

An example hierarchical representation in our formalism is illustrated in Figure 1, for the word "accelerometer."

3 Procedures

Our approach is based on a technique that exploits a context-free grammar applied to a large lexicon to aid in the preparation of a baseforms file encoding the lexicon in terms of a set of linguistically motivated subword units. The subword units, which encode syllabification and pronunciation, are initially

acrostics	-ax+ kr+ -aas t -axk +s
actualities	-aek ch+ -uw+ -ael -ax+ tf -iy+ +z
fabrications	f+ -aeb r+ -ax+ k -ey+ shaxn +z
preferences	pr+ -ehf rsyl -axn +s -axz
skepticism	sk+ -ehp t -ax+ s+ -ihz -m
striplings	str+ -ihp l+ -ihng +z

Figure 2: Sample entries from the subword lexicon.

derived automatically from a phonemic baseforms file through simple rewrite rules. The grammar is developed manually, a process that amounts to identifying all the possible ways to spell each subword unit. In an iterative procedure, parse failures are manually corrected either by modifying erroneous pronunciations or by augmenting the rules governing permissible letter sequences for the subword units. Through this process we have now converted phonemic baseforms for a lexicon of 140,000 words into the new subword units. Example entries in the baseforms file are shown in Figure 2.

Once a grammar and a large lexicon of subword baseforms are available, the next step is to create a statistical language model encoding the letter-subword mappings. We have decided to create a new set of subword units, which we call "spellnemes," combining the letter sequence and associated pronunciation into a single symbolic tag, as illustrated in Figure 3. The sequence of spellnemes associated with each word in the lexicon can easily be obtained by parsing the word, constrained by its subword realization. The spellneme sequences for each word in the lexicon are then used to train a trigram language model. Our formalism currently has 2541 unique spellnemes, on average nearly a 4-fold expansion over the number of pronunciation-based subwords.

Derivative sound-to-letter and letter-to-sound systems are straightforward. For sound-to-letter, a provided phonemic transcript is exhaustively expanded to a graph of all possible subword realizations, and subsequently into a graph of all spellnemes asso-

```
b_r<591> oo_k<547> l<617> e_t<263>
b_r<591> oo_k<547> l<617> i_n e<281>
b_r<591> oo_k<547> l<617> y_n<250>
b_r<591> oo_k<547> m<619> o_n_t<43>
```

Figure 3: Sample entries from the tagged corpus which is used to train the statistics of the n-gram language model. The numeric tags encode the associated subword unit, each of which maps to a unique phonemic sequence.

ciated with each subword. The trigram language model is applied to produce an N-best list of the top-scoring hypothesized spellneme sequences. The letter-to-sound system exhaustively expands the letter sequence into all possible spellneme sequences. After applying the trigram language model, the N-best list of spellneme sequences can be mapped to the pronunciations by concatenation of the phonemic realizations of the individual subwords.

4 Experiments on Phonebook

We imagine a two-stage speech recognition framework for a word spoken in isolation, in which the first stage uses subword units that encode only pronunciation, and produces an N-best list of hypothesized pronunciations, represented as phonemic baseforms. The second stage is tasked with hypothesizing possible spellings from the provided phonemic baseforms, and then verifying them by a match with a lexical entry. For the purposes of this paper, we assume a perfect phonemic baseform as input, and investigate the quality of the N-best list of hypothesized spellings automatically generated by the sound-to-letter system. We quantify performance by measuring the depth of the correct word in the generated N-best list.

Our experiments were conducted on a set of nearly 5000 nouns and proper nouns, a subset of the 8000 word Phonebook vocabulary that were identified as nouns using the Web site http://www.comp.lancs.ac.uk/ucrel/claws/. We selected this set of words for two reasons: (1) they contain a substantial number of nouns not included in our original training lexicon, and (2) they will allow us to conduct speech recognition experiments from the available Phonebook corpus of words spoken in isolation over the telephone.

The trigram training corpus was restricted to a subset of 55,159 entries in our original lexicon, containing the words that were tagged as nouns in Comlex. We are interested in quantifying the gap between in-vocabulary (IV) and out-of-vocabulary (OOV) words, with respect to the training corpus. We also measure the gains that can be realized through manual repair of automatically generated baseforms for training the sound-to-letter system. Thus we conducted experiments on the following four conditions:

1. Train on 55,159 nouns, test on the 3478 word IV subset of Phonebook nouns.
2. Train on 55,159 nouns, test on the 1518 OOV words in Phonebook.
3. Augment the training set with entries for the 1518 OOV words, that are obtained automatically by processing them through the letter-to-sound system. Test on the OOV subset.
4. Augment the training lexicon with manually corrected pronunciations for the OOV subset. Test on the OOV subset.

Items (3) and (4) will show us the degree to which improvements can be gained through automatic methods, once a new list of nouns becomes available, as well as how much further gain can be realized after manual correction. Automatic methods will be feasible for a dialogue system which can extract from the Web a list of nouns appropriate for the domain, but has no phonemic baseforms available for those nouns.

5 Results

Results are shown in Table 1. With an N-best list of 30, the system has a very low failure rate for all conditions. However, there is a marked difference in performance in terms of the depth of the correct answer. The mean depth is 2.07 for the OOV words, as contrasted with only 1.15 for the IV words. Fully automatic methods to improve the sound-to-letter system lead to substantial gains, reducing the mean depth to 1.54. Manual correction provides significant further gains, achieving a mean depth of 1.13, comparable to that of the original IV subset. There were two cases where an incorrect match to a lexical entry was found at a higher level in the N-best list

	Top 1	Top 2	Top 3	Top 4	Top 5	Top 30	Mean Depth	Failed
OOV	65.7%	80.7%	86.5%	90.0%	91.7%	98.4%	2.07	1.6%
plus auto	84.0%	91.6%	93.4%	94.7%	95.7%	99.0%	1.54	1.0%
plus manual	92.2%	98.0%	98.9%	99.3%	99.6%	99.9%	1.13	0.1%
IV	91.8%	97.5%	98.8%	99.3%	99.5%	100.0%	1.15	0.0%

Table 1: Percentage of words spelled correctly as a function of N-best depth for sound-to-letter experiments. See text for discussion.

than the correct match. These were the homonym pairs: carolyn/caroline and jasmine/jazzman.

Nouns that fail to appear in the top 30 can potentially still be recovered through simple spell checking methods. Using a conservative approach of allowing only a single letter insertion, substitution or deletion, and further, of requiring that the grammar could parse the corrected word under the constraints of the system's proposed subwords, we were able to recover over 60% of the failures.

6 Related Work

Many researchers have worked on letter-to-sound modeling for text-to-speech conversion (R. I. Damper and Gustafson, 1998). The topic of bi-directional phoneme-to-grapheme conversion is becoming important for application to unknown words and new word acquisition in speech understanding systems (Chung et al., 2003), although it is difficult to compare results due to different representations and data sets. In (Meng, 1996), a hierarchical approach was used for bi-directional sound-letter generation. (Rentzepopoulos and Kokkinakis, 1996) describes a hidden Markov model approach for phoneme-to-grapheme conversion, in seven European languages evaluated on a number of corpora. (Marchand and Damper, 2000) uses a fusion of data-driven and pronunciation-by-analogy methods, obtaining word accuracies of 57.7% and 69.1% for phoneme-to-grapheme and grapheme-to-phoneme experiments respectively, when evaluated on a general dictionary. (Llitjos and Black, 2001) report improvements on letter-to-sound performance on names by adding language origin features, yielding 61.72% word accuracy on 56,000 names. (Galescu and Allen, 2002) addresses bi-directional sound-letter generation using a data-driven joint n-gram method on proper nouns, yielding around 41% word accuracy for sound-to-letter and 68% word accuracy for letter-to-sound.

7 Summary and Conclusions

In this paper, we report on a new technique for reversible letter-to-sound sound-to-letter modeling, which is based on linguistic theory and statistical modeling. The system was evaluated on a set of nearly 5000 nouns from the Phonebook domain, separately for in-vocabulary and out-of-vocabulary subsets, with respect to the training corpus for the sound-to-letter system. In future work, we plan to evaluate the effectiveness of the model for automatic new word acquisition in spoken dialogue systems.

References

J. Blevins. 1995. The syllable in phonological theory. *J. Goldsmith, Ed., the Handbook of Phonological Theory. Blackwell, Oxford.*

G. Chung, S. Seneff, and C. Wang. 2003. Automatic acquisition of names using speak and spell mode in spoken dialogue systems. In *Proc. of HLT-NAACL*, Edmonton, Canada.

L. Galescu and J. Allen. 2002. Name pronunciation with a joint n-gram model for bi-directional grapheme-to-phoneme conversion. In *Proc. ICSLP*, pages 109–112, Denver, CO.

A. Font Llitjos and A. Black. 2001. Knowledge of language origin improves pronunciation accuracy of proper names. In *Proc. Eurospeech*, Aalborg, Denmark.

Y. Marchand and R. I. Damper. 2000. A multi-strategy approach to improving pronunciation by analogy. *Computational Linguistics*, 26(2):195–219.

H. Meng. 1996. Reversible letter-to-sound / sound-to-letter generation based on parsing word morphology. *Speech Computation*, 18(1):47–64.

M. J. Adamson R. I. Damper, Y. Marchand and K. Gustafson. 1998. Comparative evaluation of letter-to-sound conversion techniques for English text-to-speech synthesis. In *Proc. IWSS*, pages 53–58, Jenolan Caves, Australia.

P. Rentzepopoulos and G. K. Kokkinakis. 1996. Effi cient multilingual phoneme-to-grapheme conversion based on HMM. *Computational Linguistics*, 22(3).

Are Some Speech Recognition Errors Easier to Detect than Others?

Yongmei Shi
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, MD 21250
yshi1@umbc.edu

Lina Zhou
Department of Information Systems
University of Maryland, Baltimore County
Baltimore, MD 21250
zhoul@umbc.edu

Abstract

This study investigates whether some speech recognition (SR) errors are easier to detect and what patterns can be identified from those errors. Specifically, SR errors were examined from both non-linguistic and linguistic perspectives. The analyses of non-linguistic properties revealed that high error ratios and consecutive errors lowered the ease of error detection. The analyses of linguistic properties showed that ease of error detection was associated with changing parts-of-speech of reference words in SR errors. Additionally, syntactic relations themselves and the change of syntactic relations had impact on the ease of error detection.

1 Introduction

Speech recognition (SR) errors remain as one of the main impediment factors to the wide adoption of speech technology, especially for continuous large-vocabulary SR applications. As a result, lowering word error rate is the focus of SR research which can benefit from analyzing SR errors. SR errors have been examined from various perspectives: linguistic regularity of errors (McKoskey and Boley, 2000), the relationships between linguistic factors and SR performance (Greenberg and Chang, 2000), and the associations of prosodic features with SR errors (Hirschberg et al., 2004). However, little is understood about patterns of errors with regard to ease of detection.

Analyzing SR errors can be helpful to error detection. Skantze and Edlund (2004) conducted a user study to evaluate the effects of various features on error detection. Our study is different in that it investigates the relationships between the characteristics of SR errors and their ease of detection through an empirical user study. Given two SR systems with the same word error rates, the output of one system could be more useful if its errors are easier to detect than the other. Accordingly, SR and its error detection research could focus on addressing difficult errors by developing automatic solutions or by providing decision support to manual error detection.

2 Experiment

A laboratory experiment was carried out to evaluate humans' performance in SR error detection.

2.1 Experimental Data

Speech transcripts were extracted from a dictation corpus on daily correspondence in office environment generated using IBM ViaVoice under high-quality condition (Zhou et al., 2006).

Eight paragraphs were randomly selected from the transcripts of two task scenarios based on two criteria: recognition accuracy and paragraph length (measured by # of words). Specifically, the overall recognition accuracy (84%) and the length of a medium-sized paragraph (90 words) of the corpus were used as references.

The selected paragraphs consist of 36 sentences. Sentence lengths range from 9 to 38 words, with an average of 20. For error detection, SR output instead of references is a better base for computing

Proceedings of NAACL HLT 2007, Companion Volume, pages 157–160,
Rochester, NY, April 2007.

error rates because SR output but not reference transcripts are accessible during error detection. This may result in fewer number of deletion errors because when one SR error maps to several reference words, it is counted as one substitution error. Based on this method, there are totally 140 errors in the selected data: 104 substitution, 31 insertion, and 5 deletion errors. The error ratio, defined as the ratio of the number of errors to the number of words in output sentence, ranges from 4.76% to 61.54%.

2.2 Task and Procedure

Participants were required to read error annotation schema and sample transcripts prior to the experiment, and could attend the experiment only after they passed the test on their knowledge of the schema and SR output.

Each participant was asked to detect errors in all eight paragraphs. All sentences in the same paragraphs were presented all at once. The paragraphs were presented with different methods, including three with no additional information, three with alternative hypotheses, and two with both dictation scenario and alternative hypotheses. The sequence of paragraphs and their presentation methods were randomized for each participant.

Ten participants from a mid-sized university in the U.S. completed the study. They were all native speakers and none of them was professional editor.

3 Analysis and Discussion

In this section, we analyze the relationship between characteristics of SR errors and ease of error detection. We characterize errors with non-linguistic and linguistic properties and further break down the latter into parts-of-speech and syntactic relations.

3.1 Ease of Error Detection

The ease of detecting an error was defined as the number of participants who successfully detected the error. When computing the ease of error detection, we merged all the data by ignoring the presentation methods. The decision was made because a repeated measure ANOVA of recall failed to yield a significant effect of presentation methods ($p = n.s.$). The recall was selected because it measures the percentage of actual errors being detected and the focal interest of this study was actual errors. The average recalls of error detection of three presentation methods were very close, ranging from 72% to 75%.

The ease values fell between 0 and 10, with 0 being the least ease when all participants missed the error and 10 being the most ease when everyone found the error. To improve the power of statistical analyses, errors were separated into 3 groups using equal-height binning based on their ease values, namely 1 for low, 2 for medium, and 3 for high (see Table 1). The overall average ease value was 2.15.

Level of Ease	Ease Values	# of Errors
Low (1)	0-5	39
Medium (2)	6-8	41
High (3)	9-10	60

Table 1: Grouping of ease values

3.2 Non-linguistic Error Properties

Three non-linguistic error properties, including error ratio, word error type, and error sequence (in isolation or next to other errors) were selected to examine their relationships with ease of error detection.

Two-tailed correlation analyses of error ratio and ease of detection showed that the Pearson correlation coefficient was -0.477 ($p < 0.01$), which suggests that it is easier to detect errors in sentences with lower error ratios.

One way ANOVA failed to yield a significant effect of error type on ease of detection ($p = n.s.$). Nonetheless, mean comparisons showed that insertion errors were less easy to detect ($mean = 2.03$) than deletion errors ($mean = 2.20$) and substitution errors ($mean = 2.18$). Users may have difficulty in judging extra words.

Among the 140 errors, about half of them (i.e., 71) were next to some other errors. One way ANOVA revealed a significant effect of error sequence on ease of detection, $p < 0.05$. Specifically, isolated errors ($mean = 2.33$) are easier to detect than consecutive errors ($mean = 1.97$).

3.3 Part-Of-Speech(POS)

SR output and reference transcripts were analyzed using Brill's part-of-speech tagger (Brill, 1995). To alleviate data sparsity problem, we adopted second-

level tags such as NN and VB. The POSes of SR errors as well as POS change patterns between reference words and SR errors were analyzed.

Table 2 reports the average eases of detection for difference POSes on all the errors, substitution errors only, and insertion errors only. Deletion errors were not included because they did not appear in SR output. Only those POSes with frequency of at least 10 in all the errors were selected.

POS	All	Substitution	Insertion
NN	2.03	2.00	2.25
VB	2.30	2.41	1.67
CC	2.21	2.38	1.83
IN	2.22	2.27	2.00
DT	1.80	2.25	1.50

Table 2: Ease of detection for different POSes

It was easier to detect verbs that were misplaced than verbs that were inserted mistakenly ($p < 0.1$ in one-tailed results). This is because an additional verb may change syntactic and semantic structures of entire sentence. Similar patterns held for both CC and DT ($p < 0.1$ in one-tailed results). The less ease in detecting DT and CC when they were inserted than replaced is due in part to the fact that they play significant syntactic roles in constructing a grammatical sentence. Further, ease of detecting DT was lower than the average ease of all errors ($p < 0.1$ in one-tailed results).

Only substitution errors were applicable in POS change analysis. POS change was set to 'Y' when the POSes of an SR error and its corresponding reference word were different, and 'N' when otherwise. This resulted in 69 Ys and 35 Ns. One way ANOVA results yielded a significant effect of POS change on ease of detection ($p < 0.05$). Specifically, it was easier to detect errors that had different POSes ($mean = 2.32$) from their references than those that shared the same POSes ($mean = 1.91$). This is partly due to the requirements of semantic and even discourse information in detecting errors from the same POSes.

3.4 Syntactic Relations

Both SR output and reference transcripts were parsed using minipar (Lin, 1998), a principle-based parser that can generate a dependency parse tree for each sentence. The dependency relations between SR errors and other words in the same sentence were extracted as the syntactic relations of SR errors. The same kinds of relations were also extracted for corresponding reference words.

Three types of properties of syntactic relations were analyzed, including the number of syntactic relations, syntactic relation change, and errors' patterns of syntactic relations.

Table 3 reports descriptive statistics of ease of detection for SR errors with varying numbers of syntactic relations. The average number of syntactic relations for all errors was 1.64. Analysis results showed that it was easier to detect errors with no syntactic relations than those with one relation ($p < 0.05$). The analysis of correlation between the number of syntactic relations and the ease of detection yielded a very small Pearson correlation coefficient ($p = n.s.$). They suggest that errors that do not fit into a sentence are easy to detect. However, increasing the number of syntactic relations does not lower the ease of detection.

# of Syntactic Relations	Mean	Std Deviation	Frequency
0	2.40	0.695	35
1	1.98	0.883	51
2	2.21	0.918	19
3	2.00	0.791	17
> 3	2.22	0.808	18

Table 3: Ease of detection for numbers of relations

Same as POS change, only substitution errors were considered in syntactic relation change analysis, and the values of the syntactic changes were set similarly. By dividing the syntactic relations into head and modifier according to whether the words served as heads in the relations, we also derived syntactic changes for head and modifier relations, respectively.

Two-way ANOVA analyses of head and modifier syntactic relation changes yielded a significant interaction effect ($p < 0.05$). A post-hoc analysis revealed that, when the modifier syntactic relations were the same, it was easier to detect errors that did not cause the change of head syntactic relations than

those causing such changes ($p < 0.05$).

Table 4 reports descriptive statistics of ease of detection in terms of syntactic relations of SR errors that occurred at least 5 times. Two relations were presented in the "Syntactic Relations" column. The first one is the relation in which errors played the head role, and the second one is the relation that errors served as a modifier. "None" indicates no such relations exist.

Syntactic Relations	Mean	Std Deviation	Frequency
none none	2.40	0.695	35
none subj	2.70	0.675	10
none det	1.78	0.833	9
none punc	2.00	0.926	8
none nn	2.00	1.000	7
none pcomp-n	2.33	1.033	6
mod pcomp-n	1.20	0.447	5
none obj	1.80	0.837	5

Table 4: Ease of detection for syntactic relations

It is shown in Table 4 that it is easier to detect if an error is the subject of a verb (subj). A typical example is the "summary" in sentence "summary will have to make my travel arrangement ... ". All the participants successfully detected "summary" as an error. In contrast, "mod pcomp-n" was difficult to detect. Manual scrutinizing of the data showed that such errors were nouns that both have some other words/phrases as modifier (mod) and are nominal complements of a preposition (pcomp-n). For example, for "transaction" in sentence "I'm particularly interested in signal transaction in ... ", 80% participants failed to detect the error. It requires domain knowledge to determine the error.

4 Conclusion and Future Work

This study revealed that both high error ratio and consecutive errors increased the difficulty of error detection, which highlights the importance of SR performance. In addition, it was easier to detect SR errors when they had different POSes from corresponding reference words. Further, SR errors lacking syntactic relations were easy to detect, and changes in syntactic relations of reference words in SR errors had impact on the ease of error detection. The extracted patterns could advance SR and automatic error detection research by accounting for the ease of error detection. They could also guide the development of support systems for manual SR error correction.

This study brings up many interesting issues for future study. We plan to replicate the study with automatic error detection experiment. Additional experiments would be conducted on a larger data set to extract more robust patterns.

Acknowledgement

This work was supported by the National Science Foundation (NSF) under Grant 0328391. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

References

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4):543–565.

Steven Greenberg and Shuangyu Chang. 2000. Linguistic dissection of switchboard-corpus automatic speech recognition systems. In *Proceedings of the ISCA Workshop on Automatic Speech Recognition: Challenges for the New Millennium*.

Julia Hirschberg, Diane Litman, and Marc Swerts. 2004. Prosodic and other cues to speech recognition failures. *Speech Communication*, 43:155–175.

Dekang Lin. 1998. Dependency-based evaluation of minipar. In *Proceedings of the Workshop on the Evaluation of Parsing Systems*.

David McKoskey and Daniel Boley. 2000. Error analysis of automatic speech recognition using principal direction divisive partitioning. In *Proceedings of ECML*, pages 263–270.

Gabriel Skantze and Jens Edlund. 2004. Early error detection on word level. In *Proceedings of Robustness*.

Lina Zhou, Yongmei Shi, Dongsong Zhang, and Andrew Sears. 2006. Discovering cues to error detection in speech recogntion output: A user-centered approach. *Journal of Management of Information Systems*, 22(4):237–270.

Simultaneous Identification of Biomedical Named-Entity and Functional Relations Using Statistical Parsing Techniques *

Zhongmin Shi and **Anoop Sarkar** and **Fred Popowich**
School of Computing Science
Simon Fraser University
{zshi1,anoop,popowich}@cs.sfu.ca

Abstract

In this paper we propose a statistical parsing technique that simultaneously identifies biomedical named-entities (NEs) and extracts subcellular localization relations for bacterial proteins from the text in MEDLINE articles. We build a parser that derives both syntactic and domain-dependent semantic information and achieves an F-score of 48.4% for the relation extraction task. We then propose a semi-supervised approach that incorporates noisy automatically labeled data to improve the F-score of our parser to 83.2%. Our key contributions are: learning from noisy data, and building an annotated corpus that can benefit relation extraction research.

1 Introduction

Relation extraction from text is a step beyond Named-Entity Recognition (NER) and generally demands adequate domain knowledge to build relations among domain-specific concepts. A Biomedical Functional Relation (relation for short) states interactions among biomedical substances. In this paper we focus on one such relation: Bacterial Protein Localization (BPL), and introduce our approach for identifying BPLs from MEDLINE[1] articles.

BPL is a key functional characteristic of proteins. It is essential to the understanding of the function of different proteins and the discovery of suitable drugs, vaccines and diagnostic targets. We are collaborating with researchers in molecular biology with the goal of automatically extracting BPLs from text with BioNLP techniques, to expand their protein localization database, namely PSORTdb[2](Rey et al., 2005). Specifically, the task is to produce as output the relation tuple *BPL(BACTERIUM, PROTEIN, LOCATION)* along with source sentence and document references. The task is new to BioNLP in terms of the specific biomedical relation being sought. Therefore, we have to build annotated corpus from scratch and we are unable to use existing BioNLP shared task resources in our experiments. In this paper we extract from the text of biomedical articles a relation among: a LOCATION (one of the possible locations shown in Figure 1 for Gram+ and Gram- bacteria); a particular BACTERIUM, e.g. *E. Coli*, and a PROTEIN name, e.g. *OprF*.

(Nair and Rost, 2002) used the text taken from Swiss-Prot annotations of proteins, and trained a subcellular classifier on this data. (Hoglund et al., 2006) predicted subcellular localizations using an SVM trained on both text and protein sequence data, by assigning each protein name a vector based on terms co-occurring with the localization name for each organism. (Lu and Hunter, 2005) applied a hierarchical architecture of SVMs to predict subcellular localization by incorporating a semantic hierarchy of localization classes modeled with biological processing pathways. These approaches either ignore the actual location information in their predicted localization relations, or only focus on a small portion of eukaryotic proteins. The performance of these approaches are not comparable due to different tasks and datasets.

2 System Outline

During our system's preprocessing phase, sentences are automatically annotated with both syntactic information and domain-specific semantic information. Syntactic annotations are provided by a statistical parser (Charniak and Johnson, 2005). Domain-

*This research was partially supported by NSERC, Canada.

[1]MEDLINE is a bibliographic database of biomedical scientific articles at National Library of Medcine (NLM, http://www.nlm.nih.gov/).

[2]http://db.psort.org.

Proceedings of NAACL HLT 2007, Companion Volume, pages 161–164,
Rochester, NY, April 2007.

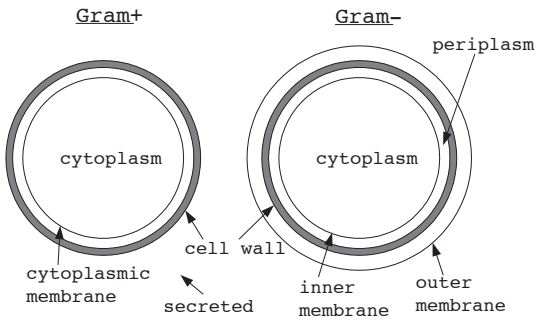


Figure 1: Illustration of possible locations of proteins with respect to the bacterial cell structure.

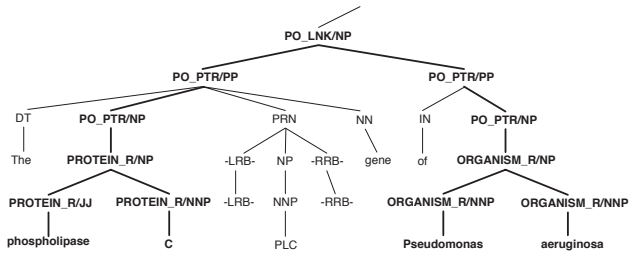


Figure 2: An example of parsing results

specific semantic information includes annotations on PROTEIN, BACTERIUM and LOCATION NEs by dictionary lookups from UMLS[3], NCBI Taxonomy[4] and SwissProt[5], and two automatic Bio-NE recognizers: MMTx[6] and Lingpipe[7].

We propose the use of a parser that simultaneously identifies NEs and extracts the BPL relations from each sentence. We define NEs to be **Relevant** to each other only if they are arguments of a BPL relation, otherwise they are defined to be **Irrelevant**. A sentence may contain multiple PROTEIN (LOCATION or ORGANISM) NEs, e.g., there are two PROTEIN NEs in the sentence below but only one, *OmpA*, is relevant. Our system aims to identify the correct BPL relation among all possible BPL tuples (candidate relations) in the sentence by only recognizing relevant NEs. Each input sentence is assumed to have at least one BPL relation.

Nine of 10 monoclonal antibodies mapped within the carboxy-terminal region of [$_{PROTEIN}$ *OprF*] *that is homologous to the* [$_{ORGANISM}$ *Escherichia coli*] [$_{LOCATION}$ *outer membrane*] *protein* [$_{PROTEIN}$ *OmpA*].

3 Statistical Syntactic and Semantic Parser

Similar to the approach in (Miller et al., 2000) and (Kulick et al., 2004), our parser integrates both syntactic and semantic annotations into a single annotation as shown in Figure 2. A lexicalized statistical parser (Bikel, 2004) is applied to the parsing task. The parse tree is decorated with two types of semantic annotations:

1) Annotations on relevant PROTEIN, BACTERIUM and LOCATION NEs. Tags are *PROTEIN_R*, *BACTERIUM_R* and *LOCATION_R* respectively.

2) Annotations on paths between relevant NEs. The lower-most node that spans both NEs is tagged as *_LNK* and all nodes along the path to the NEs are tagged as *_PTR*.

Binary relations are apparently much easier to represent on the parse tree, therefore we split the BPL ternary relation into two binary relations: BP (BACTERIUM and PROTEIN) and PL (PROTEIN and LOCATION). After capturing BP and PL relations, we will predict BPL as a fusion of BP and PL, see §4.1. In contrast to the global inference done using our generative model, heavily pipelined discriminative approaches usually have problems with error propagation. A more serious problem in a pipelined system when using syntactic parses for relation extraction is the alignment between the named entities produced by a separate system and the syntactic parses produced by the statistical parser. This alignment issue is non-trivial and we could not produce a pipelined system that dealt with this issue satisfactorily for our dataset. As a result, we did not directly compare our generative approach to a pipelined strategy.

4 Experiment Settings and Evaluations

The training and test sets are derived from a small expert-curated corpus. Table 1 lists numbers of sentences and relevant NEs in each BP/PL/BPL set.

Since the parsing results include both NE and path tags (note that we do not use any external NER system), there are two metrics to produce and evaluate PL or BP relations: **Name-only** and **Name-path** metrics. The **name-only** metric only measures **Rel-**

[3]http://www.nlm.nih.gov/research/umls/
[4]http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=Taxonomy
[5]http://www.ebi.ac.uk/swissprot/
[6]MetaMap Transfer, http://mmtx.nlm.nih.gov/
[7]http://www.alias-i.com/

	PL	BP	BPL
Training set	289 / 605	258 / 595	352 / 852
Test set	44 / 134	28 / 127	62 / 182

Table 1: Sizes of training and test sets (number of sentences / number of relevant NEs)

evant PROTEIN, BACTERIUM and LOCATION NEs (see Section 2). It does not take path annotations into account. The name-only metric is measured in terms of Precision, Recall and F-score, in which True Positive (TP) is the number of correctly identified NEs, False Positive (FP) is the number of incorrectly identified NEs and False Negative (FN) is the number of correct NEs that are not identified.

The **name-path** measures nodes being annotated as *_LNK*, *_PTR* or *_R* along the path between NEs on the parse tree, therefore it represents **confidence** of NEs being arguments of the relation. The name-path metric is a macro-average measure, which is the average performance of all sentences in data set. In measurement of the name-path metric, TP is the number of correctly annotated nodes on the path between relevant NEs. FP is the number of incorrectly annotated nodes on the path and FN is the number of correct nodes that are not identified.

4.1 Fusion of BP and PL

The BPL relation can be predicted by a fusion of BP and PL once they are extracted. Specifically, a BP and a PL that are extracted from the same sentence are merged into a BPL. The predicted BPL relations are then evaluated by the same name-only and name-path metrics as for binary relations. In the name-path metric, nodes on both PL and BP paths are counted. Note that we do not need a common protein NER to merge the BP and PL relations. E.g., for name-only evaluation, assume true BPL($B1$, $P1$, $L1$): if we predict BP($B1$,) and PL($P1$, $L2$), then *TP*=2 due to $B1$, $P1$; *FP*=1 due to $L2$; and *FN*=1 due to $P1$.

5 NER and BPL Extraction

Baseline: An intuitive method for relation extraction would assume that any sentence containing PROTEIN, ORGANISM and LOCATION NEs has the relation. We employ this method as a baseline system, in which NEs are identified by the automatic NE recognizers and dictionary lookups as introduced in §2. The system is evaluated against the test set in Table 1. Results in Table 2 show low precision for PROTEIN NER and the name-path metric.

Extraction using Supervised Parsing: We first experiment a fully supervised approach by training the parser on the BP/PL training set and evaluate on the test set (see Table 1). The name-only and name-path evaluation results in Table 2 show poor syntactic parsing annotation quality and low recall on PROTEIN NER. The major reason of these problems is the lack of training data.

Extraction using Semi-supervised Parsing: Experiments with purely supervised learning show that our generative model requires a large curated set to minimize the sparse data problem, but domain-specific annotated corpora are always rare and expensive. However, there is a huge source of unlabeled MEDLINE articles available that may meet our needs, by assuming that any sentence containing BACTERIUM, PROTEIN and LOCATION NEs has the BPL relation. We then choose such sentences from a subset of the MEDLINE database as the training data. These sentences, after being parsed and BPL relations inserted, are in fact the very noisy data when used to train the parser, since the assumed relations do not necessarily exist. The reason this noisy data works at all is probably because we can learn a preference for structural relations between entities that are close to each other in the sentence, and thus distinguish between competing relations in the same sentence. In future work, we hope to explore explicit bootstrapping from the labeled data to improve the quality of the noisy data.

Two experiments were carried out corresponding to choices of the training set: 1) noisy data only, 2) noisy data and curated training data. Evaluation results given in Table 2.

Evaluation results on the name-only metric show that, compared to supervised parsing, our semi-supervised method dramatically improves recall for NER. For instance, recall for PROTEIN NER increases from 25.0% to 81.3%; recall on BACTERIUM and LOCATION NERs increases about 30%. As for the name-path metric, the overall F-score is much higher than our fully supervised method increasing from 39.9% to 74.5%. It shows that the inclusion of curated data in the semi-

Method	Measure	Name-only Evaluation (%)					Name-Path Evaluation (%)		
		PL		BP		BPL	PL	BP	BPL
		PROT	LOC	PROT	BACT				
Baseline	**P**	42.3	78.6	41.9	81.3	40.7	27.1	38.9	31.0
	R	92.5	97.3	87.8	97.4	90.9	56.5	69.0	60.7
	F	58.0	87.0	56.7	88.6	56.2	36.6	49.8	41.0
Supervised (training data only)	**P**	66.7	87.5	66.7	72.7	76.9	45.9	41.2	43.9
	R	25.0	56.0	10.5	47.1	35.3	36.7	36.3	36.5
	F	36.4	68.3	18.2	57.1	48.4	40.8	38.6	39.9
Semi-supervised (noisy data only)	**P**	66.7	95.5	70.6	94.1	80.8	76.2	83.5	79.3
	R	84.2	80.8	80.0	84.2	81.8	67.8	72.4	67.0
	F	74.4	87.5	75.0	88.9	81.3	71.7	**77.5**	74.2
Semi-supervised (noisy data + training data)	**P**	73.9	95.5	76.5	94.1	84.8	77.0	81.1	78.7
	R	81.0	80.8	81.3	84.2	81.7	68.5	73.7	70.7
	F	**77.3**	**87.5**	**78.8**	**88.9**	**83.2**	**72.5**	77.2	**74.5**

Table 2: Name-only and name-path evaluation results. PROTEIN, LOCATION and BACTERIUM are PROT, LOC and BACT for short. The training data is the subset of curated data in Table 1.

supervised method does not improve performance much. Precision of PROTEIN NER increases 6.5% on average, while F-score of overall BPL extraction increases only slightly. We experimented with training the semi-supervised method using noisy data alone, and testing on the entire curated set, i.e., 333 and 286 sentences for BP and PL extractions respectively. Note that we do not directly train from the training set in this method, so it is still "unseen" data for this model. The F-scores of path-only and path-name metrics are 75.5% and 67.1% respectively.

6 Discussion and Future Work

In this paper we introduced a statistical parsing-based method to extract biomedical relations from MEDLINE articles. We made use of a large unlabeled data set to train our relation extraction model. Experiments show that the semi-supervised method significantly outperforms the fully supervised method with F-score increasing from 48.4% to 83.2%. We have implemented a discriminative model (Liu et al., 2007) which takes as input the examples with gold named entities and identifies BPL relations on them. In future work, we plan to let the discriminative model take the output of our parser and refine our current results further. We also plan to train a graphical model based on all extracted BP, PL and BPL relations to infer relations from multiple sentences and documents.

References

D. Bikel. 2004. A distributional analysis of a lexicalized statistical parsing model. In *Proc. of EMNLP '04*, pages 182–189.

E. Charniak and M. Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proc. of ACL '05*, pages 173–180.

A. Hoglund, T. Blum, S. Brady, P. Donnes, J. Miguel, M. Rocheford, O. Kohlbacher, and H. Shatkay. 2006. Significantly improved prediction of subcellular localization by integrating text and protein sequence data. In *Proc. of PSB '06*, volume 11, pages 16–27.

S. Kulick, A. Bies, M. Libeman, M. Mandel, R. McDonald, M. Palmer, A. Schein, and L. Ungar. 2004. Integrated annotation for biomedical information extraction. In *Proc. of HLT/NAACL '04*, pages 61–68, Boston, May.

Y. Liu, Z. Shi, and A. Sarkar. 2007. Exploiting rich syntactic information for relation extraction from biomedical articles. In *NAACL-HLT '07, poster track*, Rochester, NY, April.

Z. Lu and L. Hunter. 2005. Go molecular function terms are predictive of subcellular localization. In *Proc. of PSB '05*, volume 10, pages 151–161.

S. Miller, H. Fox, L. Ramshaw, and R. Weischedel. 2000. A novel use of statistical parsing to extract information from text. In *Proc. of NAACL '06*, pages 226–233.

R. Nair and B. Rost. 2002. Inferring subcellular localization through automated lexical analysis. In *Bioinformatics*, volume 18, pages 78–86.

S. Rey, M. Acab, J. Gardy, M. Laird, K. deFays, C. Lambert, and F. Brinkman. 2005. Psortdb: A database of subcellular localizations for bacteria. *Nucleic Acids Research*, 33(D):164–168.

Virtual Evidence for Training Speech Recognizers using Partially Labeled data

Amarnag Subramanya
Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500
asubram@u.washington.edu

Jeff Bilmes
Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500
bilmes@ee.washington.edu

Abstract

Collecting supervised training data for automatic speech recognition (ASR) systems is both time consuming and expensive. In this paper we use the notion of virtual evidence in a graphical-model based system to reduce the amount of supervisory training data required for sequence learning tasks. We apply this approach to a TIMIT phone recognition system, and show that our VE-based training scheme can, relative to a baseline trained with the full segmentation, yield similar results with only 15.3% of the frames labeled (keeping the number of utterances fixed).

1 Introduction

Current state-of-the-art speech recognizers use thousands of hours of training data, collected from a large number of speakers with various backgrounds in order to make the models more robust. It is well known that one of the simplest ways of improving the accuracy of a recognizer is to increase the amount of training data. Moreover, speech recognition systems can benefit from being trained on hand-transcribed data where all the appropriate word level segmentations (i.e., the exact time of the word boundaries) are known. However, with increasing amounts of raw speech data being made available, it is both time consuming and expensive to accurately segment every word for every given sentence. Moreover, for languages for which only a small amount of training data is available, it can be expensive and challenging to annotate with precise word transcriptions – the researcher may have no choice but to use partially erroneous training data.

There are a number of different ways to label data used to train a speech recognizer. First, the most expensive case (from an annotation perspective) is fully supervised training, where both word sequences and time segmentations are completely specified[1]. A second case is most commonly used in speech recognition systems, where only the word sequences of utterances are given, but their precise segmentations are unknown. A third case falls under the realm of semi-supervised approaches. As one possible example, a previously trained recognizer is used to generate transcripts for unlabeled data, which are then used to re-train the recognizer based on some measure of recognizer confidence (Lamel et al., 2002).

The above cases do not exhaust the set of possible training scenarios. In this paper, we show how the notion of virtual evidence (VE) (Pearl, 1988) may be used to obtain the benefits of data with time segmentations but using only partially labeled data. Our method lies somewhere between the first and second cases above. This general framework has been successfully applied in the past to the activity recognition domain (Subramanya et al., 2006). Here we make use of the TIMIT phone recognition task as an example to show how VE may be used to deal with partially labeled speech training data. To the best of our knowledge, this paper presents the first system to express training uncertainty using VE in the speech domain.

2 Baseline System

Figure 1 shows two consecutive time slices of a dynamic Bayesian network (DBN) designed for con-

[1]This does not imply that all variables are observed during training. While the inter-word segmentations are known, the model is not given information about intra-word segmentations.

Proceedings of NAACL HLT 2007, Companion Volume, pages 165–168,
Rochester, NY, April 2007.

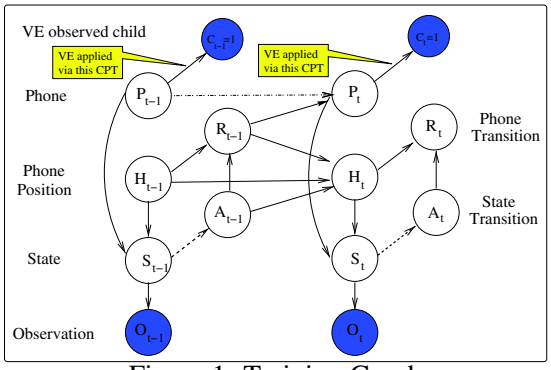


Figure 1: Training Graph.

text independent (CI) phone recognition. All observed variables are shaded, deterministic dependences are depicted using solid black lines, value specific dependences are shown using a dot-dash lines, and random dependencies are represented using dashed lines. In this paper, given any random variable (rv) X, x denotes a particular value of that rv, D_X is the domain of X ($x \in D_X$), and $|D_X|$ represents its cardinality.

In the above model, P_t is the rv representing the phone variable, H_t models the current position within a phone, S_t is the state, O_t the acoustic observations, A_t and R_t indicate state and phone transitions respectively. Here, $D_{X_t} = D_{X_{t-1}}$, $\forall t, \forall X$. In our implementation here, $D_{H_t}, D_{A_t} \in \{0, 1, 2\}$, $D_{R_t} \in \{0, 1\}$. Also $\delta\{c_1, \ldots, c_n\}$ is an indicator function that turns on when all the conditions $\{c_1, \ldots, c_n\}$ are true (i.e. a conjunction over all the conditions). The distribution for H_t is given by $p(h_t|h_{t-1}, r_{t-1}, a_{t-1}) = \delta_{\{h_t=0, r_{t-1}=1\}} + \delta_{\{h_t=a_{t-1}+h_{t-1}, r_{t-1}=0\}}$, which implies that we always start a phone with $H_t = 0$. We allow skips in each phone model, and A_t=0, indicates no transition, A_t=1 implies you transition to the next state, A_t=2 causes a state to skip ($H_{t+1} = H_t + 2$). As the TIMIT corpus provides phone level segmentations, P_t is observed during training. However, for reasons that will become clear in the next section, we treat P_t as hidden but make it the parent of a rv C_t, with, $p(c_t = 1|p_t) = \delta_{l_t=p_t}$ where l_t is obtained from the transcriptions ($l_t \in D_{P_t}$). The above formulation has exactly the same effect as making P_t observed and setting it equal to l_t (Bilmes, 2004). Additional details on other CPTs in this model may be found in (Bilmes and Bartels, 2005). We provide more details on the baseline system in section 4.1.

Our main reason for choosing the TIMIT phone recognition task is that TIMIT includes both sequence and segment transcriptions (something rare

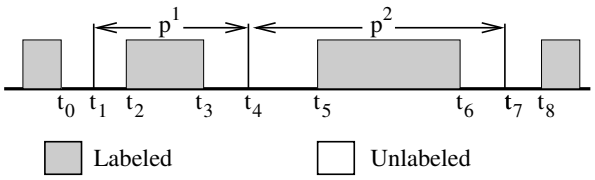


Figure 2: Illustration showing our rendition of Virtual Evidence.

for LVCSR corpora such as Switchboard and Fisher). This means that we can compare against a model that has been trained fully supervised. It is also well known that context-dependent (CD) models outperform CI models for the TIMIT phone recognition task (Glass et al., 1996). We used CI models primarily for the rapid experimental turnaround time and since it still provides a reasonable test-bed for evaluating new ideas. We do note, however, that our baseline CI system is competitive with recently published CD systems (Wang and Fosler-Lussier, 2006), albeit which uses many fewer components per mixture (see Section 4.1).

3 Soft-supervised Learning With VE

Given a joint distribution over n variables $p(x_1, \ldots, x_n)$, "evidence" simply means that one of the variables (w.l.o.g. x_1) is known. We denote this by $\bar{x}_1$, so the probability distribution becomes $p(\bar{x}_1, \ldots, x_n)$ (no longer a function of x_1). Any configuration of the variables where $x_1 \neq \bar{x}_1$ is never considered. We can mimic this behavior by introducing a new virtual child variable c into the joint distribution that is always observed to be one (so $c = 1$), and have c interact only with x_1 via the CPT $p(c = 1|x_1) = \delta_{x_1=\bar{x}_1}$. Therefore, $\sum_{x_1} p(c = 1, x_1, \ldots, x_n) = p(\bar{x}_1, \ldots, x_n)$. Now consider setting $p(c = 1|x_1) = f(x_1)$, where $f()$ is an arbitrary non-negative function. With this, different treatment can be given to different assignments to x_1, but unlike hard evidence, we are not insisting on only one particular value. This represents the general notion of VE. In a certain sense, the notion of VE is similar to the prior distribution in Bayesian inference, but it is different in that VE expresses preferences over combinations of values of random variables whereas a Bayesian prior expresses preferences over combinations of model parameter values. For a more information on VE, see (Bilmes, 2004; Pearl, 1988).

VE can in fact be used when accurate phone level segmentations are not available. Consider the illustration in Figure 2. As shown, t_1 and t_4 are the

start and end times respectively for phone p^1, while t_4 and t_7 are the start and end times for phone p^2. When the start and end times for each phone are given, we have information about the identity of the phone that produced each and every observation. The general training scenario in most large vocabulary speech recognition systems, however, does not have access to these starting/ending times, and they are trained knowing only the sequence of phone labels (e.g., that p^2 follows p^1).

Consider a new transcription based on Figure 2, where we know that p^1 ended at some time $t_3 \leq t_4$ and that p^2 started at sometime $t_5 > t_4$. In the region between t_3 and t_5 we have no information on the identity of the phone variable for each acoustic frame, except that it is either p^1 or p^2. A similar case occurs at the start of phone p^1 and the end of phone p^2. The above information can be used in our model (Figure 1) in the following way (here given only for $t_2 \leq t \leq t_6$): $p(C_t = 1|p_t) = \delta_{\{p_t=p^1, t_2 \leq t \leq t_3\}} + \delta_{\{p_t=p^2, t_5 \leq t \leq t_6\}} + f_t(p^1)\delta_{\{p_t=p^1, t_3 \leq t \leq t_5\}} + g_t(p^2)\delta_{\{p_t=p^2, t_3 \leq t \leq t_5\}}$. Here $f_t(p^1)$ and $g_t(p^2)$ represent our relative beliefs at time t in whether the value of P_t is either p^1 or p^2. It is important to highlight that rather than the absolute values of these functions, it is their relative values that have an effect on inference (Bilmes, 2004). There are number of different ways of choosing these functions. First, we can set $f_t(p^1) = g_t(p^2) = \alpha, \alpha > 0$. This encodes our uncertainty regarding the identity of the phone in this region while still forcing it to be either p^1 or p^2, and equal preference is given for both (referred to as "uniform over two phones"). Alternatively, other functions could take into account the fact that, in the frames 'close' to t_3, it is more likely to be p^1, whereas in the frames 'close' to t_5, it is more likely to be p^2. This can be represented by using a decreasing function for $f_t(p^1)$ and an increasing function for $g_t(p^2)$ (for example linearly increasing or decreasing with time).

As more frames are dropped around transitions (e.g., as $t_3 - t_2$ decreases), we use lesser amounts of labeled data. In an extreme situation, we can drop all the labels ($t_3 < t_2$) to recover the case where only sequence and not segment information is available. Alternatively, we can have $t_3 = t_2 + 1$, which means that only one frame is labeled for every phone in an utterance — all other frames of a phone are left untranscribed. From the perspective of a transcriber, this simulates the task of going through an utterance and identifying only one frame that belongs to each particular phone without having to identify the phone boundary. In contrast to the task of determining the phone boundary, identifying one frame per word unit is much simpler, less prone to error or disagreement, and less costly (Greenberg, 1995).

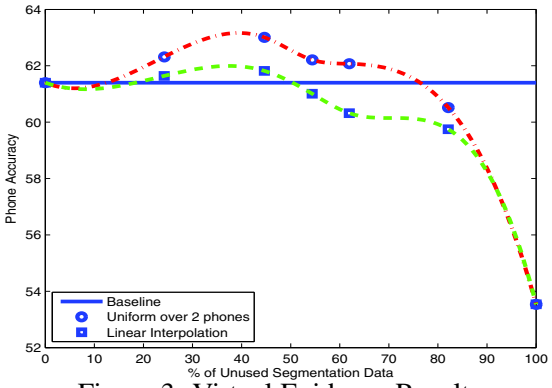


Figure 3: Virtual Evidence Results

4 Experimental Results

4.1 Baseline System

We trained a baseline TIMIT phone recognition system that made full use of all phone level segmentations (the fully supervised case). To obtain the acoustic observations, the signal was first preemphasized ($\alpha = 0.97$) and then windowed using a Hamming window of size 25ms at 100Hz. We then extracted MFCC's from these windowed features. Deltas and double deltas were appended to the above observation vector. Each phone is modeled using 3 states, and 64 Gaussians per state. We follow the standard practice of building models for 48 different phones and then mapping them down to 39 phones for scoring purposes (Halberstadt and Glass, 1997). The decoding DBN graph is similar to the training graph (Figure 1) except that the variable C_t is removed when decoding. We test on the NIST Core test set (Glass et al., 1996). All results reported in this paper were obtained by computing the string edit (Levenshtein) distance between the hypothesis and the reference. All models in this paper were implemented using the Graphical Models Toolkit (GMTK) (Bilmes and Bartels, 2005).

4.2 VE Based Training and Results

We tested various cases of VE-based training by varying the amount of "dropped" frame labels on either side of the transition (the dropped labels became the unlabeled frames of Figure 2). We did this until there was only one frame left labeled for every phone. Moreover, in each of the above cases, we tested a number of different functions to gener-

ate the VE scores (see section 3). The results of our VE experiments are shown in Figure 3. The curves were obtained by fitting a cubic spline to the points shown in the figure. The phone accuracy (PA) of our baseline system (trained in a fully supervised manner) is 61.4%. If the total number of frames in the training set is N_T, and we drop labels on N frames, the amount of unused data is given by $U = \frac{N}{N_T} * 100$ (the x-axis in the figure). Thus $U = 0\%$ is the fully supervised case, whereas $U = 100\%$ corresponds to using only the sequence information. Dropping the label for one frame on either side of every phone transition yielded $U = 24.5\%$.

It can be seen that in the case of both "uniform over 2 phones" and linear interpolation, the PA actually improves when we drop a small number ($\leq$ 5 frames) of frames on either side of the transition. This seems to suggest that there might be some inherent errors in the frame level labels near the phone transitions. The points on the plot at U=84.7% correspond to using a single labeled frame per phone in every utterance in the training set (average phone length in TIMIT is about 7 frames). The PA of the system using a single label per phone is 60.52%. In this case, we also used a trapezoidal function defined as follows: if $t = t_i$ were the labeled frames for phone p^1, then $f_t(p^1) = 1, t_i - 1 \leq t \leq t_i + 1$, and a linear interpolation function for the other values t during the transition to generate the VE weights. This system yielded a PA of 61.29% (baseline accuracy 61.4%). We should highlight that even though this system used only 15.3% of the labels used by the baseline, the results were similar! The figure also shows the PA of the system that used only the sequence information was about 53% (compare against baseline accuracy of 61.4%). This lends evidence to the claim that training recognizers using data with time segmentation information can lead to improved performance.

Given the procedure we used to drop the frames around transitions, the single labeled frame for every phone is usually located on or around the midpoint of the phone. This however cannot be guaranteed if a transcriber is asked to randomly label one frame per phone. To simulate such a situation, we randomly choose one frame to be labeled for every phone in the utterance. We then trained this system using the "uniform over 2 phones" technique and tested it on the NIST core test set. This experiment was repeated 10 times, and the PA averaged over the 10 trails was found to be 60.5% (standard deviation 0.402), thus showing the robustness of our technique even for less carefully labeled data.

5 Discussion

In this paper we have shown how VE can be used to train a TIMIT phone recognition system using partially labeled data. The performance of this system is not significantly worse than the baseline that makes use of all the labels. Further, though this method of data transcription is only slightly more time consuming that sequence labeling, it yeilds significant gains in performance (53% v/s 60.5%). The results also show that even in the presence of fully labaled data, allowing for uncertainity at the transitions during training can be beneficial for ASR performance. It should however be pointed out that while phone recognition accuracy is not always a good predictor of word accuracy, we still expect that our method will ultimately generalize to word accuracy as well, assuming we have access to a corpus where at least one frame of each word has been labeled with the word identity. This work was supported by an ONR MURI grant, No. N000140510388.

References

[Bilmes and Bartels2005] J. Bilmes and C. Bartels. 2005. Graphical model architectures for speech recognition. *IEEE Signal Processing Magazine*, 22(5):89–100, September.

[Bilmes2004] J. Bilmes. 2004. On soft evidence in Bayesian networks. Technical Report UWEETR-2004-0016, University of Washington, Dept. of EE.

[Glass et al.1996] J. Glass, J. Chang, and M. McCandless. 1996. A probabilistic framework for feature-based speech recognition. In *Proc. ICSLP '96*, volume 4, Philadelphia, PA.

[Greenberg1995] S Greenberg. 1995. The Switchboard transcription project. Technical report, The Johns Hopkins University (CLSP) Summer Research Workshop.

[Halberstadt and Glass1997] A. K. Halberstadt and J. R. Glass. 1997. Heterogeneous acoustic measurements for phonetic classification. In *Proc. Eurospeech '97*, pages 401–404, Rhodes, Greece.

[Lamel et al.2002] L. Lamel, J. Gauvian, and G. Adda. 2002. Lightly supervised and unsupervised acoustic model training. *Computer Speech and Language*.

[Pearl1988] J. Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc.

[Subramanya et al.2006] A. Subramanya, A. Raj, J. Bilmes, and D. Fox. 2006. Recognizing activities and spatial context using wearable sensors. In *Proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*.

[Wang and Fosler-Lussier2006] Y. Wang and E. Fosler-Lussier. 2006. Integrating phonetic boundary discrimination explicity into HMM systems. In *Proc. of the Interspeech*.

A High Accuracy Method for Semi-supervised Information Extraction

Stephen Tratz
Pacific Northwest National Laboratory
Richland, WA 99352
stephen.tratz@pnl.gov

Antonio Sanfilippo
Pacific Northwest National Laboratory
Richland, WA 99352
antonio.sanfilippo@pnl.gov

Abstract

Customization to specific domains of discourse and/or user requirements is one of the greatest challenges for today's Information Extraction (IE) systems. While demonstrably effective, both rule-based and supervised machine learning approaches to IE customization pose too high a burden on the user. Semi-supervised learning approaches may in principle offer a more resource effective solution but are still insufficiently accurate to grant realistic application. We demonstrate that this limitation can be overcome by integrating fully-supervised learning techniques within a semi-supervised IE approach, without increasing resource requirements.

1 Introduction

Customization to specific discourse domains and/or user requirements is one of the greatest challenges for today's Information Extraction (IE) systems. While demonstrably effective, both rule-based and supervised machine learning approaches to IE customization require a substantial development effort. For example, Aone and Ramos-Santacruz (2000) present a rule-based IE system which handles 100 types of relations and events. Building such a system requires the manual construction of numerous extraction patterns supported by customized ontologies. Soderland (1999) uses supervised learning to induce a set of rules from hand-tagged training examples. While Sonderland suggests that the human effort can be reduced by interleaving learning and manual annotation activities, the creation of training data remains an onerous task.

To reduce the knowledge engineering burden on the user in constructing and porting an IE system, unsupervised learning has been utilized, e.g. Riloff (1996), Yangarber et al. (2000), and Sekine (2006). Banko et al. (2007) present a self-supervised system that aims to avoid the manual IE customization problem by extracting all possible relations of interest from text. Stevenson and Greenwood (2005) propose a weakly supervised approach to *sentence filtering* that uses semantic similarity and bootstrapping to acquire IE patterns. Stevenson's and Greenwood's approach provides some of the best available results in weakly supervised IE to date, with 0.58 F-measure. While very good, an F-measure of 0.58 does not provide sufficient reliability to grant use in a production system.

In this paper, we show that it is possible to provide a significant improvement over Stevenson's and Greenwood's results, without increasing resource requirements, by integrating fully-supervised learning techniques within a weakly supervised IE approach.

1.1 Learning Algorithm

Our method is modeled on the approach developed by Stevenson and Greenwood (2005) but uses a different technique for ranking candidate patterns. Stevenson's and Greenwood's algorithm takes as data inputs a small set of initial seed patterns and a corpus of documents, and uses any of several semantic similarity measures (Resnik, 1995; Jiang and Conrath, 1997; Patwardhan et al., 2003) to iteratively identify patterns in the document corpus

Proceedings of NAACL HLT 2007, Companion Volume, pages 169–172,
Rochester, NY, April 2007.

that bear a strong resemblance to the seed patterns. After each iteration, the top-ranking candidate patterns are added to the seed patterns and removed from the corpus. Our approach differs from that of Stevenson and Greenwood in that we use a supervised classifier to rank candidate patterns. This grants our system greater robustness and flexibility because the weight of classification features can be automatically determined within a supervised classification approach.

In building supervised classifiers to rank candidate patterns at each iteration, we use both positive and negative training examples. Instead of creating manually annotated training examples, we follow an active learning approach where training examples are automatically chosen by ranking candidate patterns in terms of cosine similarity with the seed patterns. More specifically, we select patterns that have the lowest similarity with seed patterns to be the negative training examples. We hypothesized that these negative examples would contain many of the uninformative features occurring throughout the corpus and that using these examples would enable the classifier to determine that these features would not be useful.

The pattern learning approach we propose includes the following steps.

1. An unannotated corpus is required as input. For each sentence, a set of features is extracted. This information becomes S_{cand}, the set of all candidate patterns.
2. The user defines a set of seed patterns, S_{seed}. These patterns contain features expected to be found in a relevant sentence.
3. The cosine measure is used to determine the distance between the patterns in S_{seed} and S_{cand}. The patterns in S_{cand} are then ordered by their lowest distance to a member of S_{seed}.
4. The α highest ranked patterns in S_{cand} are added to S_{pos}, the set of positive training examples.
5. S_{seed} and S_{acc} are added to S_{pos}. S_{neg}, the set of negative training examples is constructed from β+iteration*γ of the lowest ranked patterns in S_{cand}. Then, a maximum entropy classifier is built using S_{pos} and S_{neg} as training data.
6. The classifier is used to score each pattern in S_{cand}. S_{cand} is then sorted by these scores.
7. The top δ patterns in S_{cand} are added to S_{acc} and removed from S_{cand}.
8. If a suitable stopping point has been reached, the process ends. Otherwise, S_{pos} and S_{neg} are emptied and the process continues at step 6.

We set α to 5, β to 20, γ to 15, δ to 5, and used the following linguistic processing tools: (1) the OpenNLP library (opennlp.sourceforge.net) for sentence splitting and named-entity recognition, and (2) Connexor for syntactic parsing (Tapanainen and Järvinen, 1997). For the classifier, we used the OpenNLP MaxEnt implementation (maxent.sourceforge.net) of the maximum entropy classification algorithm (Berger et al. 1996). We used the MUC-6 data set as the testing ground for our proposed approach.

1.2 Description of Features Used

Stevenson and Greenwood (2005) use subject-verb-object triples for their features. We use a richer feature set. Our system can easily accommodate more features because we let the maximum entropy classifier determine the weight for the features. Stevenson's and Greenwood's approach determines weights using semantic similarity and would require significant changes to take into account various other features, especially those for which a WordNet (Fellbaum, 1998) similarity score is not available.

We use single tokens, token combinations, and semantic information to inform our IE pattern extraction system. Lexical items marked by the named-entity recognition system as PERSON or ORGANIZATION are replaced with 'person' and 'organization', respectively. Number tokens are replaced with 'numeric'. Single Token Features include:

- All words in the sentence and all hypernyms of the first sense of the word with attached part-of-speech
- All words in the sentence with attached dependency
- The verb base of each nominalization and the verb's first sense hypernyms are included.

Token Combinations include:

- All bigrams from the sentence
- All subject-object pairs
- All parent-child pairs from the parse tree

- A specially marked copy of the parent-child pairs where the main verb is the parent.

We also added semantic features indicating if a PERSON or ORGANIZATION was detected within the sentence boundaries. Table1 provides an example where a simple sentence is mapped into the set of features we have just described.

```
Alan G. Spoon, 42, will succeed
Mr. Graham as president of the
company.
```

Single Token Features
With attached dependencies: attr:person, subj:person, mod:numeric, v-ch:will, main:succeed, obj:person, copred:as, pcomp:president, mod:of, det:the, pcomp:company *With part-of-speech tags:* n:person, v:succeed, v:will, dt:the, n:company, n:institution, n:social_group, n:group, n:organization, n:person, n:president, n:executive, n:corporate_executive, n:administrator, n:head, n:leader, n:organism, n:living_thing, n:object, n:entity, num:numeric, abbr:person, prp:as, prp:of, v:control, v:declare, v:decree, v:express, v:ordain, v:preside, v:state
Token Combinations
Bigrams: person+comma, comma+numeric, numeric+comma, comma+will, will+succeed, succeed+person, person+as, as+president, president+of, of+the, the+company *Subject Object Pairs:* sop:person+person *Parent-Child Pairs:* pc:person+person, pc:person+numeric, pc:will+person, pc:succeed+will, pc:succeed+person, pc:succeed+as, pc:as+president, pc:president+of, pc:of+company, pc:company+the *Main Verb Parent-Child Pairs:* mvpc:succeed+person, mvpc:succeed+will, mvpc:succeed+as
Semantic Features
hasOrganization, hasPerson

Table 1: Feature representation of a simple sentence.

The seeds we used are adapted from the seed patterns employed by Stevenson and Greenwood. As shown in Table 2, only a subset of the features described above is used in the seed patterns.

2 Evaluation

We used the document collection which was initially developed for the Sixth Message Understanding Conference (MUC-6) as ground truth data set to evaluate our approach. The MUC-6 corpus (www.ldc.upenn.edu) is composed of 100 Wall Street Journal documents written during 1993 and 1994. Our task was to detect sentences which included management succession patterns, such as those shown in Table 2.

1: subj:organization, main:appoint, obj:person, hasPerson, hasOrganization
2: subj:organization, main:elect, obj:person, hasOrganization, hasPerson
3: subj:organization, main:promote, obj:person, hasOrganization, hasPerson
4: subj:organization, main:name, obj:person, hasOrganization, hasPerson
5: subj:person, main:resign, hasPerson
6: subj:person, main:depart, hasPerson
7: subj:person, main:quit, hasPerson

Table 2: Feature representation of seed patterns.

The version of the MUC-6 corpus produced by Soderland (1999) provided us with a specification of succession patterns at the sentence level, but as shown in Table 3 did not include the source text. We reconstructed the original text by automatically aligning the succession patterns in the sentence structures in Soderland's version of the MUC-6 corpus with the sentences in the original MUC-6 corpus. This alignment produced a set of 1581 sentences, of which 134 contained succession patterns.

```
@S[
  {SUBJ  @CN[ FOX ]CN }
  {VB  NAMED @NAM }
  {OBJ  @PN[ LUCILLE S. SALHANY ]PN , @PS[
CHAIRMAN ]PS OF @CN[ FOX INC. ]CN 'S
TELEVISION PRODUCTION ARM , }
  {REL_V  TO SUCCEED @SUCCEED HIM . }
]@S 9301060123-5
@@TAGS Succession {PersonIn @PN[ LUCILLE S.
SALHANY ]PN}+ {Post @PS[ CHAIRMAN ]PS}+
{Org @CN[ FOX INC. ]CN}_ @@COVERED_BY
@@ENDTAGS
```

Table 3: Data sample from Soderland test set.

As shown in Figure 1, our best score of 0.688 F-measure was obtained on the 36th iteration; at the end of this iteration, our algorithm selected 180 sentences including 108 of the sentences that contained succession patterns. This is a significant improvement over the 0.58 F-measure score

reported by Stevenson and Greenwood (2005) for the same task. The use of a supervised classification approach to the ranking of candidate patterns with a richer feature set were the two determinant factors in achieving such improvement.

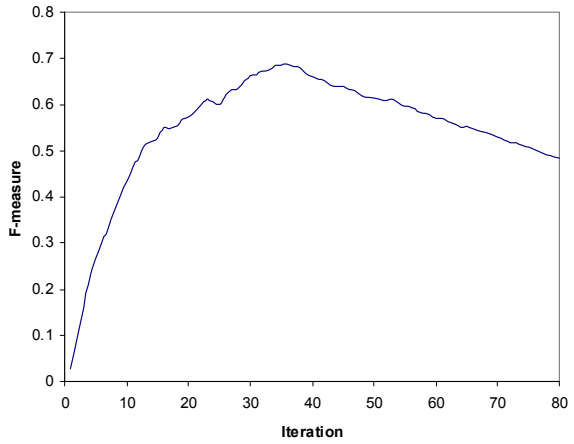


Figure 1: Evaluation results with MUC-6 data.

3 Conclusions

Our results show a substantial improvement over previous efforts in weakly supervised IE methods, suggesting that weakly supervised methods can be made to rival rule-based or fully supervised approaches both in resource effectiveness and accuracy. We plan to verify the strength of our approach evaluating against other ground truth data sets. We also plan to detail how the various features in our classification model contribute to ranking of candidate patterns. An additional area of envisioned improvement regards the use of a random sub selection of negative candidate patterns as training samples to counteract the presence of sentence fragments among low-ranking candidate patterns. Finally, we intend to evaluate the benefit of having a human in the loop in the first few iterations to filter out patterns chosen by the system.

References

C. Aone and M. Ramos-Santacruz. 2000. REES: A Large-Scale Relation and Event Extraction System, pages 76-83, In *Proceedings of the 6th Applied Natural Language Processing Conference (ANLP 2000)*, Seattle.

M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. 2007. Open Information Extraction from the Web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*. Hyderabad, India.

A. Berger, S. Della Pietra and V. Della Pietra (1996) A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics,* volume 22, number 1, pages 39-71.

C. Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database and some of its Applications*. MIT Press, Cambridge, MA.

J. Jiang and D. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. *In Proceedings of the 10th International Conference on Research in Computational Linguistics*, Taiwan.

S. Patwardhan, S. Banerjee, and T. Pederson. 2003. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the 4th International Conferences on Intelligent Text Processing and Computational Linguistics*, pages 241-257, Mexico City.

P. Resnik. 1995. Using Information Content to evaluate Semantic Similarity in a Taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 448-452, Montreal, Canada.

E. Riloff and R. Jones. 1999. Learning Dictionaries for Information Extraction by Multi-level Bootstrapping. In *Proceedings of the 16th National Conference on Artificial Intelligence*. Orlando, Florida.

S. Sekine. 2006. On-Demand Information Extraction. In Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions. Sydney, Australia.

S. Soderland. 1999. Learning Information Extraction Rules for Semi-structured and free text. *Machine Learning*, 31(1-3):233-272.

M. Stevenson and M. A. Greenwood. 2005. A Semantic Approach to IE Pattern Induction. In *Proceedings of the 43rd Annual Meeting of the ACL (ACL 05)*, Ann Arbor, Michigan.

P. Tapanainen and Timo Järvinen. 1997. A non-projective dependency parser. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 64–71, Washington D.C. Association for Computational Linguistics.

R. Yangarber, R. Grishman, P. Tapanainen, and S. Huttunen. 2000. Automatic acquisition of domain knowledge for information extraction. In *Proceedings of the 18th International Conference of Computational Linguistics (COLING 2002)*, Taipei.

The Effects of Word Prediction on Communication Rate for AAC

Keith Trnka, Debra Yarrington, John McCaw, and Kathleen F. McCoy
Department of Computer and Information Sciences
University of Delaware Newark, DE 19716
trnka,yarringt,mccaw,mccoy@cis.udel.edu

Christopher Pennington
AgoraNet, Inc.
314 East Main Street, Suite 1
Newark, DE 19711
penningt@agora-net.com

Abstract

Individuals using an Augmentative and Alternative Communication (AAC) device communicate at less than 10% of the speed of "traditional" speech, creating a large communication gap. In this user study, we compare the communication rate of pseudo-impaired individuals using two different word prediction algorithms and a system without word prediction. Our results show that word prediction can increase AAC communication rate and that more accurate predictions significantly improve communication rate.

1 Introduction

Communication is a significant quality-of-life issue for individuals with severe speech impairments. The field of Augmentative and Alternative Communication (AAC) is concerned with mitigating communication barriers that would otherwise isolate individuals from society. Most high-tech AAC devices provide the user with an electronic letter and word board to input messages which are output via speech synthesis. However, even with substantial user interface optimization, communication rate is often less than 10 words per minute (Newell et al., 1998) as compared to about 150-200 words per minute for unimpaired speech.

One way to improve communication rate is to decrease the number of keys entered to form a message. Word prediction is an application of language modeling to allowing the user to access words they may be spelling at a cost of one keystroke. Many commercial AAC devices use word prediction, such as PRC's Pathfinder™, Dynavox Technology's Dynavox 4™, and Saltillo's ChatPC™.

Although word prediction is used in AAC devices, researchers have questioned whether it actually increases communication rate (Venkatagiri, 1993; Koester and Levine, 1997; Anson et al., 2004). These works note the additional cognitive demands and cost of using word prediction in conjunction with a letter-by-letter interface, such as the need to shift the focus of attention to the prediction list, the time to scan the prediction list, and the cognitive effort required for making decisions about the predicted words. Obviously the design of the particular interface (e.g., the ease of using word prediction) will affect these results. In addition, these studies used a single, simplistic method of generating predictions, and this may also be responsible for some of their results.

In contrast, other researchers (Lesher and Higginbotham, 2005; Li and Hirst, 2005; Trnka et al., 2006) have continued to investigate various improvements to language modeling for word prediction in order to save the user more keystrokes. Newer methods such as topic modeling yield statistically significant keystroke savings over previous methods. However, the question remains as to whether improvements in prediction methods translate to an enhanced communication rate. We hypothesize that it will.

In this paper we study (1) whether a word prediction interface increases communication rate over

Proceedings of NAACL HLT 2007, Companion Volume, pages 173–176,
Rochester, NY, April 2007.

letter-by-letter typing when a reasonable prediction method is employed and (2) whether an advanced word prediction method increases communication rate over a basic word prediction method to a degree greater than that afforded by the difference in theoretical keystroke savings between the two methods. We expect that the communication rate gain due to the better word prediction method will exceed the gains from the poorer system. Our reasons for this expectation has to do with not only users wasting time scanning lists that do not contain the desired word, but also the tendency for a user to give up on such a system (i.e., choosing to ignore the predictions) and thus missing the predicted word even if it does appear in the list. Validating these hypotheses will motivate continued improvements in word prediction methods for increased communication rate.

The target population of our research is adult AAC users without significant cognitive impairments. Including actual AAC users in the study poses several significant complications, perhaps the largest of which concerns the user interface. AAC devices vary significantly in the physical interfaces available, in accordance with the variety of physical abilities of AAC users. This diversity has caused different word prediction interfaces to be developed for each physical interface. Moreover, it would be impossible to mimic our word prediction layout in a consistent fashion on all of the major AAC devices used. Because of this, we conducted this pilot study using subjects that are pseudo-impaired: the subjects have no motor impairments but we have simulated a motor impairment by providing an interface that emulates the communication rate of a typical AAC user. Future work includes the verification of the results using a smaller number of actual AAC users.

2 Approach

The purpose of the study was to measure the effects of word prediction methods on communication rate. To this end, the interface used for text entry was optimized for ease-of-use and kept constant across trials. Subjects were asked to enter text on a touchscreen monitor using Wivik™, an on-screen keyboard. Because we wanted to simulate AAC users with motor impairments, we programmed a 1.5 second delay between a key press and its registration in the system. The artificial impairment gave the subjects the same incentive to use word prediction that AAC users face every day, whereas users with fine motor control tend to ignore word prediction (e.g., in common word processing software). The delay slows the input rate of our subjects down to a rate more typical of AAC users (about 8-10 words per minute).

Seventeen adult, native speakers of English with no visual, cognitive, or motor impairments participated in the study. These subjects were asked to type in three different excerpts from held-out data of the Switchboard corpus on three different days.[1] In each of these sessions, a different prediction method was used and the order of prediction methods was randomized across subjects. Keystrokes and predictions were logged and then post-processed to compute the words produced per minute, seconds per keystroke, and keystroke savings, among other statistics.

2.1 Independent variable: prediction methods

The independent variable in our study is the method of text entry used: (1) letter-by-letter typing using the Wivik keyboard with *no word prediction*, (2) letter-by-letter typing augmented with word predictions produced by a *basic prediction method*, (3) letter-by-letter typing augmented with word predictions produced by an *advanced prediction method*.

Basic prediction generates predictions from the combination of a recency model of the text entered so far in conjunction with a large word list. The recency model is given priority in generating predictions. This model is similar to language models used in AAC devices with the exception that many devices use a unigram model in lieu of a word list.

Advanced prediction generates predictions on the basis of a trigram model with backoff. A special unigram model is used for the first word in each sentence. This language model is constructed from the transcribed telephone conversations of the Switchboard corpus. If the prediction list isn't filled from this model's predictions, then predictions are selected from a recency model and then a word list, as in the basic prediction method.

[1] Switchboard was chosen because our prediction models were trained using another portion of the corpus. A copy task was chosen for more controlled experimental conditions.

	Adv. prediction	Basic prediction	No prediction
Words per minute (wpm)	8.09	5.50	5.06
Time (seconds)	1316s	1808s	2030s
Seconds per keystroke (spk)	2.92s	2.58s	2.28s
Keystroke savings (ks)	50.3%	18.2%	-
Potential keystroke savings (pks)	55.2%	25.0%	-
Prediction utilization (pru)	90.9%	73.3%	-

Figure 1: Average statistics for each method.

3 Results

Once the data was collected, we post-processed the logs and accumulated statistics. Average values for each method are shown in Figure 1 and comparative values are shown in Figure 2.

3.1 Communication rate (output rate)

The overall average words per minute and task completion time for each method is shown in Figure 1, and Figure 2 shows comparative data for the three methods. As hypothesized, advanced prediction was found to be significantly faster than basic prediction and basic prediction was found to be significantly faster than no prediction ($\alpha = 0.01$). For example, users produced 59.9% more words per minute using advanced prediction compared to no prediction. Advanced prediction was 44.4% faster than basic prediction but basic prediction was only 10.1% faster than no prediction.

Additionally, the relative task completion time is shown in Figure 2. The copy tasks with advanced prediction were completed in 64.5% of the time it took to complete without word prediction. The trend shown with relative task completion time reinforces the trends shown with words per minute – advanced prediction offers a large speedup over no prediction and basic prediction, but basic prediction offers a much smaller increase over no prediction.

Our results show that basic word prediction significantly boosts communication rate and that advanced word prediction substantially increases communication rate beyond basic prediction.

3.2 Input rate (seconds per keystroke)

Figures 1 and 2 indicate that there were significant differences (at $\alpha = 0.01$) in the methods in terms of the rate at which keys were pressed. In particular, while overall communication rate was significantly faster with advanced prediction, users took 0.641 seconds longer for each key press from using advanced prediction compared to entry without prediction. Similarly, users spent 0.345s longer to enter each key using advanced as opposed to basic prediction and basic prediction required more time per keystroke than no prediction. The slower input rate can be attributed to the additional demands of searching through a prediction list and making a decision about selecting a word from that list over continuing to type letters.

3.3 Keystroke savings / prediction utilization

The difference between the potential keystroke savings offered by advanced and basic prediction is substantial: 55.2% vs. 25.0%, as shown in Figure 1. Accordingly, the actual keystroke savings that users realized under each prediction method shows a wide separation: 50.3% for advanced and 18.2% for basic. The keystroke savings that users of basic prediction achieved seems quite a bit lower than the potential keystroke savings offered by the predictions. In other words, the prediction utilization of basic prediction was much lower than that of advanced prediction. Comparative analysis shows a 17.1% improvement in prediction utilization from advanced over basic prediction.

4 Discussion

The results show that communication rate increased despite the decreased input rate due to a large reduction in the amount of input required (high keystroke savings). In the past, researchers have noted that the cognitive load of using word prediction was considerable, so that the keystroke savings of word pre-

	Adv. over None	Adv. over Basic	Basic over None
Relative task completion time	0.645[1]	0.701[1]	0.919[1]
Words per minute (wpm)	59.9% faster[2]	44.4% faster[2]	10.1% faster[2]
Seconds per keystroke (spk)	0.641s[2]	0.345s[2]	0.286s[2]
Prediction utilization (pru)		17.1%[2]	

Figure 2: Average per-subject improvements. ([1] Significance not tested. [2] Significant at $\alpha = 0.01$.)

diction was outweighed by the overhead of using it. However, we have shown that despite significant cognitive load, the reduction in keystroke savings dominates the effect on output rate.

In contrast to earlier studies, our basic method showed a significantly improved communication rate over no prediction. One reason for this could be the intuitiveness of our user interface. A second reason could be related to the consistency of the basic prediction method. In particular, at least some subjects using the basic prediction method learned to scan the prediction list when the desired word was recently used and mentioned it in the exit survey. At other times they simply ignored the prediction list and proceeded with letter-by-letter typing. This behavior would also explain why the input was significantly slower with the advanced method over the basic method – users found that scanning the prediction list more often was worth the added effort. This also explains the significant difference in prediction utilization between the methods.

The relationship between keystroke savings and communication rate is a trend of increasing rate enhancement with increasingly accurate prediction methods. Improved prediction methods offer greater potential keystroke savings to users and users see increased keystroke savings in practice. Additionally, users rely on better predictions more and thus lose less of the potential keystroke savings offered by the method. We expect that keystroke savings will see substantial increases from improved potential keystroke savings until prediction utilization is closer to 100%.

5 Conclusions

Word prediction in an experimental AAC device with simulated AAC users significantly enhances communication rate. The difference between an advanced and basic prediction method demonstrates that further improvements in language modeling for word prediction are likely to appreciably increase communication rate. Therefore, further research in improving word prediction is likely to have an important impact on quality-of-life for AAC users. We plan to improve word prediction and validate these results using AAC users as future work.

Acknowledgments

This work was supported by US Department of Education grant H113G040051.

References

Denis Anson, Penni Moist, Mary Przywars, Heather Wells, Heather Saylor, and Hantz Maxime. 2004. The effects of word completion and word prediction on typing rates using on-screen keyboards. *Assistive Technology*, 18.

Heidi Horstmann Koester and Simon P. Levine. 1997. Keystroke-level models for user performance with word prediction. *Augmentative and Alternative Communication*, 13:239–257, December.

Gregory W. Lesher and D. Jeffery Higginbotham. 2005. Using web content to enhance augmentative communication. In *Proceedings of CSUN 2005*.

Jianhua Li and Graeme Hirst. 2005. Semantic knowledge in word completion. In *ASSETS '05*, pages 121–128.

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

Keith Trnka, Debra Yarrington, Kathleen F. McCoy, and Christopher A. Pennington. 2006. Topic modeling in fringe word prediction for aac. In *IUI '06*, pages 276–278.

Horabail S. Venkatagiri. 1993. Efficiency of lexical prediction as a communication acceleration technique. *Augmentative and Alternative Communication*, 9:161–167, September.

Language Modeling for Determiner Selection

Jenine Turner and Eugene Charniak
Department of Computer Science
Brown Laboratory for Linguistic Information Processing (BLLIP)
Brown University
Providence, RI 02912
{jenine|ec}@cs.brown.edu

Abstract

We present a method for automatic determiner selection, based on an existing language model. We train on the Penn Treebank and also use additional data from the North American News Text Corpus. Our results are a significant improvement over previous best.

1 Introduction

Determiner placement (choosing if a noun phrase needs a determiner, and if so, which one) is a non-trivial problem in several language processing tasks. While context beyond that of the current sentence can sometimes be necessary, native speakers of languages with determiners can select determiners quite well for most NPs. Native speakers of languages without determiners have a much more difficult time.

Automating determiner selection is helpful in several applications. A determiner selection program can aid in Machine Translation of determiner-free languages (by adding determiners after the text has been translated), correct English text written by non-native speakers (Lee, 2004), and choose determiners for text generation programs.

Early work on determiner selection focuses on rule-based systems (Gawronska, 1990; Murata and Nagao, 1993; Bond and Ogura, 1994; Heine, 1998). Knight and Chander (1994) use decision trees to choose between *the* and *a/an*, ignoring NPs with no determiner, and achieve 78% accuracy on their Wall Street Journal corpus. (Deciding between *a* and *an* is a trivial postprocessing step.)

Minnen et al. (2000) use a memory-based learner (Daelemans et al., 2000) to choose determiners of base noun phrases. They choose between no determiner (hencefore *null*), *the*, and *a/an*. They use syntactic features (head of the NP, part-of-speech tag of the head of the NP, functional tag of the head of the NP, category of the constituent embedding the NP, and functional tag of the constituent embedding the NP), whether the head is a mass or count noun and semantic classes of the head of the NP (Ikehara et al., 1991). They report 83.58% accuracy.

In this paper, we use the Charniak language model (Charniak, 2001) for determiner selection. Our approach significantly improves upon the work of Minnen et al. (2000). We also use additional automatically parsed data from the North American News Text Corpus (Graff, 1995), further improving our results.

2 The Immediate-Head Parsing Model

The language model we use is described in (Charniak, 2001). It is based upon a parser that, for a sentence s, tries to find the parse π defined as:

$$\arg max_{\pi} p(\pi \mid s) = \arg max_{\pi} p(\pi, s) \quad (1)$$

The parser can be turned into a language model $p(s)$ describing the probability distribution over all possible strings s in the language, by considering all parses π of s:

$$p(s) = \sum_{\pi} p(\pi, s) \quad (2)$$

Proceedings of NAACL HLT 2007, Companion Volume, pages 177–180,
Rochester, NY, April 2007.

Here $p(\pi, s)$ is zero if the yield of $\pi \neq s$.

The parsing model assigns a probability to a parse π by a top-down process. For each constituent c in π it first guesses the pre-terminal of c, $t(c)$ (t for "tag"), then the lexical head of c, $h(c)$, and then the expansion of c into further constituents $e(c)$. Thus the probability of a parse is given by the equation

$$\begin{aligned} p(\pi) &= \prod_{c \in \pi} p(t(c) \mid l(c), H(c)) \\ &\quad \cdot\ p(h(c) \mid t(c), l(c), H(c)) \\ &\quad \cdot\ p(e(c) \mid l(c), t(c), h(c), H(c)) \end{aligned}$$

where $l(c)$ is the label of c (e.g., whether it is a noun phrase NP, verb phrase, etc.) and $H(c)$ is the relevant history of c — information outside c deemed important in determining the probability in question. $H(c)$ approximately consists of the label, head, and head-part-of-speech for the parent of c: $m(c), i(c)$, and $u(c)$ respectively and also a secondary head (e.g., in "Monday Night Football" Monday would be conditioned on both the head of the noun-phrase "Football" and the secondary head "Night").

It is usually clear to which constituent we are referring and we omit the (c) in, e.g., $h(c)$. In this notation the above equation takes the following form:

$$\begin{aligned} p(\pi) &= \prod_{c \in \pi} p(t \mid l, m, u, i) \cdot p(h \mid t, l, m, u, i) \\ &\quad \cdot\ p(e \mid l, t, h, m, u). \end{aligned} \tag{3}$$

Next we describe how we assign a probability to the expansion e of a constituent. We break up a traditional probabilistic context-free grammar (PCFG) rule into a left-hand side with a label $l(c)$ drawn from the non-terminal symbols of our grammar, and a right-hand side that is a sequence of one or more such symbols. For each expansion we distinguish one of the right-hand side labels as the "middle" or "head" symbol $M(c)$. $M(c)$ is the constituent from which the head lexical item h is obtained according to deterministic rules that pick the head of a constituent from among the heads of its children. To the left of M is a sequence of one or more left labels $L_i(c)$ including the special termination symbol $\triangle$, which indicates that there are no more symbols to the left. We do the same for the labels to the right, $R_i(c)$. Thus, an expansion $e(c)$ looks like:

$$l \rightarrow \triangle L_m ... L_1 M R_1 ... R_n \triangle. \tag{4}$$

The expansion is generated first by guessing M, then in order L_1 through L_{m+1} (= $\triangle$), and then, R_1 through R_{n+1}.

Let us turn to how this works in the case of determiner recovery. Consider a noun-phrase, which, missing a possible determiner, is simply "FBI." The language model is interested in the probability of the strings "the FBI," "a/an FBI" and "FBI." The version with the highest probability will dictate the determiner, or lack thereof. So, consider (most of) the probability calculation for the answer "the FBI:"

$$\begin{aligned} &p(\text{NNP} \mid H) \cdot p(\text{FBI} \mid \text{NNP}, H) \\ &\quad \cdot\ p(\text{det} \mid \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ p(\triangle \mid \text{det}, \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ p(\text{the} \mid \text{det}, \text{FBI}, \text{NNP}, H) \end{aligned} \tag{5}$$

Of these, the first two terms, the probability that the head will be an NNP (a singular proper noun) and the probability that it will be "FBI", are shared by all three competitors, *null*, *the*, and *a/an*. These terms can therefore be ignored when we only wish to identify the competitor with the highest probability. The next two probabilities state that the noun-phrase contains a determiner to the left of "FBI" and that the determiner is the last constituent of the left-hand side. The last of the probabilities states that the determiner in question is *the*. Ignoring the first two probabilities, the critical probabilities for "the FBI" are:

$$\begin{aligned} &p(\text{det} \mid \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ p(\triangle \mid \text{det}, \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ p(\text{the} \mid \text{det}, \text{FBI}, \text{NNP}, H) \end{aligned} \tag{6}$$

Conversely, to evaluate the probability of the noun-phrase "FBI" — i.e., no determiner, we evaluate:

$$p(\triangle \mid \text{FBI}, \text{NNP}, H) \tag{7}$$

We ask the probability of the NP stopping immediately to the left of "FBI." For "a/an FBI" we evaluate:

$$\begin{aligned} &p(\text{det} \mid \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ p(\triangle \mid \text{det}, \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ (p(\text{a} \mid \text{det}, \text{FBI}, \text{NNP}, H) + \\ &\qquad p(\text{an} \mid \text{det}, \text{FBI}, \text{NNP}, H)) \end{aligned} \tag{8}$$

Test Data	Method	Accuracy
leave-one-out	Minnen et al.	83.58%
	Language Model (LM)	86.74%
tenfold on development	LM	84.72%
	LM trained on WSJ + 3 million words of NANC	85.83%
	LM trained on WSJ + 10 million words of NANC	86.36%
	LM trained on WSJ + 20 million words of NANC	86.64%
tenfold on test	LM trained on WSJ + 20 million words of NANC	86.63%

Table 1: Results of classification

This equation is very similar to Equation 6 (the equation for "the FBI", except the term for the probability of *the* is replaced by the sum of the probabilities for *a* and *an*.

To choose between *null*, *the*, or *a/an*, the language model in effect constructs Equations 6, 7 and 8 and we pick the one that has the highest probability.

2.1 Training the model

As with (Minnen et al., 2000), we train the language model on the Penn Treebank (Marcus et al., 1993). As far as we know, language modeling always improves with additional training data, so we add data from the North American News Text Corpus (NANC) (Graff, 1995) automatically parsed with the Charniak parser (McClosky et al., 2006) to train our language model on up to 20 million additional words.

3 Results and Discussion

The best results of Minnen et al. (2000) are using leave-one-out cross-validation. We also test our language model using leave-one-out cross-validation on the Penn Treebank (Marcus et al., 1993) (WSJ), giving us 86.74% accuracy (see Table 1).

Leave-one-out cross-validation does not make sense in this case. When choosing determiners, we can train a language model on similar data, but not on other NPs in the article. Therefore, for the rest of our tests, we use tenfold cross-validation. The difference between leave-one-out and tenfold cross-validation is due to the co-occurrence of NPs within an article. Church (2000) shows that a word appears with much higher probability when seen elsewhere in an article. Thus, a rare NP might be unseen in tenfold cross-validation, but seen in leave-one-out.

For each of our sets in tenfold cross validation, we use 80% of the Penn Treebank for training, 10% for development, and 10% for testing. The divisions occur at article boundaries. On our development set with tenfold cross-validation, we get 84.72% accuracy using the language model (Table 1).

As expected, we achieve significant improvement when adding NANC data over training on data from the Penn Treebank alone (Table 1). With 20 million additional words, we seem to be approaching an upper bound on the language model features. We obtain improvement despite the fact that the parses were automatic, but there may have been errors in determiner selection due to parsing error.

Table 2 gives "error" examples. Some errors are wrong (either grammatically or yielding a significantly different interpretation), but some "incorrect" answers are reasonable possibilities. Furthermore, even all the text of the article is not enough for classification at times. In particular note Example 5, where unless you know whether IBM was *the* world leader or simply one of the world leaders at the time of the article, no additional context would help.

4 Conclusions and Future Work

With the Charniak (Charniak, 2001) language model, our results exceed those of the previous best (Minnen et al., 2000) on the determiner selection task. This shows the benefits of the language model features in determining the most grammatical determiner to use in a noun phrase. Such a language model looks at much of the structure in individual sentences, but there may be additional features that could improve performance. There is a high rate of ambiguity for many of the misclassified sentences.

The success of using a state-of-the-art language

Guess	Correct	Sentence
the	*null*	(1) The computers were crude by **today's** standards.
null	*the*	(2) In addition, **the Apple II** was an affordable $1,298. (3) Highway officials insist **the ornamental railings** on older bridges aren't strong enough to prevent vehicles from crashing through.
a/an	*the*	(4) **The new carrier** can tote as many as four cups at once. (5) IBM, **the world leader** in computers, didn't offer its first PC until August 1981 as many other companies entered the market.
the	*a/an*	(6) In addition, the Apple II was **an affordable $1,298**. (7) "The primary purpose of **a railing** is to contain a vehicle and not to provide a scenic view," says Jack White, a planner with the Indiana Highway Department.
a/an	*null*	(8) Crude as they were, these early PCs triggered **explosive product development** in desktop models for the home and office.

Table 2: Examples of "errors"

model in determiner selection also suggests that one would be helpful in making other decisions in the surface realization stage of text generation. This is an avenue worth exploring.

Acknowledgements

This work was supported by NSF PIRE grant OISE-0530118. We would also like to thank the BLLIP team for their comments.

References

Francis Bond and Kentaro Ogura. 1994. Countability and number in Japanese-to-English machine translation. In *15th International Conference on Computational Linguistics*, pages 32–38.

Eugene Charniak. 2001. Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*. The Association for Computational Linguistics.

Kenneth Church. 2000. Empirical estimates of adaptation: The chance of Two Noriegas is closer to $p/2$ than p^2. In *Proceedings of COLING-2000*.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2000. TiMBL: Tilburg memory based learner, version 3.0, reference guide. ILK Technical Report ILK-0001, ILK, Tilburg University, The Netherlands.

Barbara Gawronska. 1990. Translation great problem. In *Proceedings of the 13th International Conference on Computational Linguistics*.

David Graff. 1995. *North American News Text Corpus*. Linguistic Data Consortium. LDC95T21.

Julia E. Heine. 1998. Definiteness predictions for Japanese noun phrases. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 519–525.

Satoru Ikehara, Satoship Shirai, Akio Yokoo, and Hiromi Nakaiwa. 1991. Toward an MT system without pre-editing - effects of new methods in ALT-J/E. In *Third Machine Translation Summit*, pages 101–106.

Kevin Knight and Ishwar Chander. 1994. Automated postediting of documents. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 779–784.

John Lee. 2004. Automatic article restoration. In *Proceedings of the 2004 NAACL Conference Student Research Workshop*, pages 195–200.

Michell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of HLT-NAACL 2006*.

Guido Minnen, Francis Bond, and Ann Copestake. 2000. Memory-based learning for article generation. In *Proceedings of the Fourth Conference on Computational Natural Language Learning and of the Second Learning Language in Logic Workshop*, pages 43–48.

Masaki Murata and Makoto Nagao. 1993. Determination of referential property and number of nouns in Japanese sentences for machine translation into English. In *Fifth International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 218–225.

Entity Extraction is a Boring Solved Problem – or is it?

Marc Vilain
The MITRE Corporation
Burlington Rd
Bedford MA 01730 USA
mbv@mitre.org

Jennifer Su
The MITRE Corporation *and*
Cornell University
Ithaca NY 14853 USA
jfs29@cornell.edu

Suzi Lubar
The MITRE Corporation
Burlington Rd
Bedford MA 01730 USA
slubar@mitre.org

Abstract

This paper presents empirical results that contradict the prevailing opinion that entity extraction is a boring solved problem. In particular, we consider data sets that resemble familiar MUC/ACE data, and report surprisingly poor performance for both commercial and research systems. We then give an error analysis that suggests research challenges for entity extraction that are neither boring nor solved.

1 Background

Entity extraction or named entity recognition, as it is sometimes called, is a known and familiar problem. Named entity (NE) tagging has been the subject of numerous shared-task evaluations, including the seminal MUC 6, MUC 7 and MET evaluations, the CoNLL shared task, the SIGHAN bake-offs, and the ACE evaluations. With this track record, and with commercial vendors now selling named-entity tagging for a fee, many naturally consider entity extraction to be an essentially solved problem. The present paper challenges this view.

The main issue, as we see it, is transfer: NE taggers developed for a specific corpus tend not to perform well on other data sets. Kosseim and Poibeau (2001), for one, show that the informal language of email or speech transcriptions befuddles taggers built for journalistic text. Minkov *et al* (2005) further explore the systematic differences between journalistic and informal texts, training separate taggers for each text source of interest.

Because named entity taggers are so strongly based on surface features, it isn't surprising to observe poor tagger transfer across texts with significantly different styles or with unrelated content. In this paper, we report on the more surprising result that transfer issues arise even for texts with closely aligned content or closely aligned styles.

In particular, we consider a range of primarily business-related texts that are, on the face of it, close in style and/or substance to the journalistic stories in existing NE data sets, MUC 6 in particular. We thus would have expected these texts to support good transfer performance from taggers configured to the MUC task. Instead, we found the same kinds of performance drops as Kosseim and Poibeau had noted for informal texts. Our aim here is to shed light on the how and why of this.

2 Scope of the present study

We begin with a disclaimer. Our goal is not so much to present new technical solutions to NE recognition, as to draw attention to those aspects of the problem that remain unsolved. We cover two main thrusts: (i) a black-box evaluation of several NE taggers (commercial and research systems); and (ii) an error analysis of system performance.

2.1 Evaluation data

Our evaluation data set contains three distinct sections. The largest component consists of publicly-available financial reports filed with the Securities and Exchange Commission (SEC), in particular the 2003 forms 10-K filed by eight Fortune 500 companies. These corporate annual reports share the same subject matter as much business news: sales, profits, acquisitions, business strategies and the like. They take, however, a more technical slant and are rich in accounting jargon. They are also longer, ranging in our study from 22 to 54 pages.

Proceedings of NAACL HLT 2007, Companion Volume, pages 181–184,
Rochester, NY, April 2007.

Pocahontas	Rule-based
Belle	Rule-based
Jasmine	Statistical, HMM, MUC-trained
Mulan	Statistical, CRF, MUC-trained
Ariel	Rule-based, 10-K tuning

Table 1: system pseudonyms.

Preliminary exploration with our own MUC 6 tagger showed these SEC filings to be particularly hard to tag. Because their sheer length and technical emphasis seemed implicated in this poor performance, we assembled a second corpus of forty Web-hosted business stories from such news providers as MS-NBC, *CNN Money*, and *Motley Fool*. These stories focus on the same eight companies as our 10-K data set, but are shorter and less technical, thus allowing us to isolate length and technicality as factors in tagging business texts.

The final portion of our test set consists of ten news stories that were selected to closely match the kind of data used in past MUC evaluations. They were drawn from the New York Times (NYT) and Wall Street Journal (WSJ) on-line editions, and focus on current events, thus providing one more comparable dimension of evaluation.[1]

2.2 Evaluated systems

Five systems participated in our study, representing a range of commercial tools and research prototypes. Two of these are state-of-the-art hand-built systems based on rule/pattern interpreters. Two are open-source statistical systems, one based on HMMs, and the other on CRFs; both were trained on the MUC 6 data set. The final system is our own legacy MUC-style tagger, noted as *Ariel* in Table 1. Except as noted below, all the systems were run out of the box, with no adaptation to the data.

License and privacy concerns prevent us from identifying all the systems; instead this paper reports most results anonymously, using the names of Disney heroines as system pseudonyms. We have, however exposed the identity of our own system out of fairness, as it benefited somewhat from earlier tuning to SEC forms 10-K.

2.3 Evaluation method

We attempted to replicate the procedure used in the MUC evaluations, extending it only as required by the characteristics of the taggers. The test data were formatted as in MUC 6, and where SGML markup ran afoul of system I/O characteristics, we remapped the data manually, resolving, *e.g.*, crossing tags that may have strayed into the output.

To provide scores that could be compared with the MUC evaluations, we created MUC6-compliant answer keys (Sundheim, 1995), and remapped system output to this standard. We removed system responses that were considered non-taggable in MUC (*e.g.*, URLs) and conflated fine-grained distinctions not made in MUC (*e.g.*, remapping *country* tags to *location*). Scores were assessed with the venerable MUC scorer, which provides partial credit for system responses that match the key in type but not extent, or vice-versa. The scorer also provides a full error analysis, separately characterizing each error in a system response.

3 Findings

Table 2, overleaf, presents our overall findings, aggregated across the three primary entity types: *person*, *organization*, and *location* (the ENAMEX types in the MUC standard). We generally did not measure the MUC TIMEX (*dates*, *times*) and NUMEX types (*moneys*, *percents*) because: (i) neither of the statistical systems generate them; (ii) those systems that do generate them tend to do well; (iii) they are overwhelmingly more frequent in the SEC data than in news, thus skewing results. For completeness' sake, however, Table 2 does provide all-entity news scores in parentheses for those systems that happened to generate the full set of MUC-6 entities.

Turning now to actual performance measurements, Table 2 does not present an especially pretty picture. Aside from two systems' runs on the MUC-like current events, all the scores are substantially below those obtained by competitive MUC systems, which typically reached F scores in the mid-90s, with a high of F=96 at MUC-6.

SEC. The worst performances were turned in for SEC filings, as shown in the first block of rows in Table 2. While precision is generally poor, recall is even worse. One reason for this is the very frequent rightwards shortenings of company names (*e.g.*, from *3M Corporation* to *the Corporation*), in contrast to the leftwards shortening (*e.g., 3M*) favored in news texts. *Ariel* had been tuned to tag all these cases, but the other systems only tagged a scattershot fraction. To isolate the contribution of

[1] We will make the non-copyrighted part of our corpus (the 10-Ks) available to other researchers.

	Pocahontas	Belle	Jasmine	Mulan	Ariel
SEC filings	R=58	R=28	R=50	R=50	R=71
	P=65	P=52	P=43	P=56	P=79
	F=61.1	F=36.4	F=42.7	F=52.6	F=74.5
SEC filings, *"the Corp."* optional	R=71	R=36	R=55	R=60	R=71
	P=65	P=52	P=40	P=56	P=79
	F=68.0	F=42.8	F=46.2	F=57.9	F=74.7
Business news	R=80 (82)	R=64 (69)	R=76	R=65	R=71 (75)
	P=80 (79)	P=86 (83)	P=63	P=74	P=74 (75)
	F=80.1 (81)	F=73.5 (75)	F=69.1	F=69.2	F=72.3 (75)
Current events (MUC-like)	R=94 (94)	R=59 (63)	R=79	R=79	R=89 (91)
	P=94 (93)	P=82 (80)	P=70	P=92	P=91 (92)
	F=94.3 (94)	F=68.5 (71)	F=74.5	F=84.9	F=90.4 (92)

Table 2: aggregated extraction scores, *ENAMEX* only, unless parenthesized (in parens = all entities).

these cases to system recall error, we recalculated the scores by making the cases optional. The scorer removes missing optional responses from the recall denominator, and as expected recall improved; see the second block in Table 2.

Business news. The most consistent performance across systems was achieved with business news, with scores ranging in F=69-80. This is a huge improvement over the gaping F=36-75 range we saw with SEC filings (F=43-75 with optional short names). This confirms that length and financial jargon are implicated in the poor performance on forms 10-K. Nonetheless, these improved scores are still 15-20 points lower than the better MUC scores. Is business language just hard to tag?

MUC-like news. Our attempt to replicate the MUC evaluation data yields an equivocal answer. Two systems (*Pocahontas* and *Ariel*) achieved MUC6-level scores; it may not be coincidental that both are next-generation versions of systems that participated at MUC. Of the other systems, MUC-trained *Mulan* also showed substantial improvement going from business news to current events.

While it is good news that three of the systems that were explicitly trained on MUC (manually or statistically) did well on MUC-like data, it is disquieting to see how poorly this training generalized to other news texts.

4 Factors affecting performance

A finer analysis of our three data sets helps triangulate the factors leading to the systematic performance differences shown in Table 2.

Prevalence of organizations. One factor especially stands out: as Table 3 shows, organizations are twice as prevalent in the business sources as in the MUC-like data. As organization scores generally trail scores for persons and locations (Table 4), this partly explains why business texts are hard.

Kinds of organizations. But that does not explain it all. The profiles in Figure 1 show that current events favor government/quasi-government names (*e.g., "Congress," "Hamas"*). They are less linguistically productive than the corporate and quasi-corporate names in business texts, and so are more amenable to being explicitly listed in name gazetteers. Florian *et al* (2003) note the effectiveness of gazetteers for tagging the CoNLL corpus.

Editorial standards. Our business news data reflect a growing portion of Web-hosted texts that relax the journalistic editorial rules of traditional news sources such as the NYT or WSJ. For instance, our data show the same frequent omission of corporate designators (*e.g. "inc."*) that Kosseim noted in informal text. Whereas news sources of record will generally mention a company's designator at least once in a story, our business data frequently fail to do so at all, thus removing a key name-tagging cue. By tracing the *Ariel* rule base, we found that the absence of any designator was implicated in 81% of the system's recall error for organization names.

Length. Name taggers often overcome this kind of missing evidence by second-passing a text, propagating name mentions identified in the first

	SEC	Business	MUC
Org	70%	65%	29%
Per	9%	23%	35%
Loc	21%	12%	36%

Table 3: Relative distribution of entity types

	Poca.	Belle	Jasm.	Mul.	Ariel
S org	F=62	F=10	F=46	F=53	F=83
opt	F=74	F=14	F=52	F=61	F=83
S per	F=75	F=65	F=49	F=64	F=60
S loc	F=79	F=77	F=49	F=74	F=78
B org	F=77	F=72	F=70	F=63	F=66
B. per	F=90	F=85	F=70	F=69	F=79
B. loc	F=78	F=76	F=59	F=75	F=73
M org	F=90	F=58	F=48	F=80	F=80
M per	F=99	F=90	F=84	F=81	F=92
M loc	F=98	F=81	F=74	F=89	F=95

Table 4: Type subcores (S=SEC, B=biz., M=MUC)

pass to matching but undetected mentions (Mikheev, 1999). This strategy runs foul, though, when the first pass produces precision errors, as these too can get propagated. Document length is implicated in this through the greater cumulative likelihood of making an error on the first pass and of finding a mention that matches the error on the second pass.

Quasi-names and non-names. A final factor that especially afflicts the Forms 10-K is the similarity of names and non-names. Non-taggable product names (*"AMD Athlon"*) often look like legitimate subsidiaries, while valid operating divisions (*"Health Care"*) are often hard to distinguish from generic designations of market segments.

5 Implications for further research.

What surprised us most in conducting this study was to find so obvious a transfer gap among what appear to be very similar text sources. We were also surprised by the involvement in this of relaxed editorial standards around seeming trivia (like the keyword *"inc."*) This suggests, for one, that current techniques remain too dependent on skin-deep word co-occurrence features. It also suggests that the editorially pristine news texts used in so much NE research may be atypically easy to tag.

While name-tagging programs may struggle with editorially informal texts, the absence of surface contextual cues poses no noticeable challenge to human readers. What cues are left, and there are many, are semantic in nature: predicate-argument structure, selectional restrictions, organization of the lexicon, *etc.* Recent efforts to create common propositional banks and lexical ontologies may thus have much to offer. Indeed, current research in these areas is just beginning to trickle down to the name-tagging problem (Mohit & Hwa, 2005).

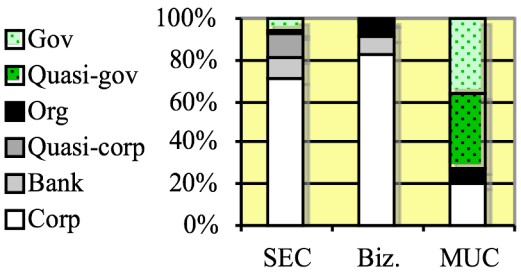


Figure 1. Kinds of organization

Another key issue is ensuring tagging coherency at the whole-document level. This might help alleviate the kind of error propagation with dual-pass strategies that particularly afflicts long documents. Recent applications of statistical co-reference models are beginning to show promise (Finkel *et al*, 2005; Ji & Grishman, 2005).

Lastly, we can see this whole study as a particular challenge case for transfer learning, and indeed such work as Sutton and McCallum's (2005) has looked at the name-tagging task from a transfer learning standpoint.

It may thus be that today's exciting emerging work in "unsolved" areas – semantics, reference, and learning – could come to play a key role in what is sometimes maligned as yesterday's boring solved problem.

References

Finkel J R, Grenager T, Manning C (2005). Incorporating non-local information into information extraction systems by Gibbs sampling, *Proc ACL*, Ann Arbor.

Florian R, Ittycheriah A, Jing H, Zhang T (2003). Named entity recognition through classifier combination, *Proc CoNLL*, Edmonton.

Ji H, Grishman R (2005). Improving name tagging by reference resolution and relation detection, *Proc ACL.*

Kosseim L, Poibeau T (2001). Extraction de noms propres à partir de textes variés. *Proc. TALN,* Toulouse.

Mikheev A, Moens M, Grover C (1999). Named entity recognition without gazetteers. *Proc. EACL*, Bergen.

Minkov E, Wang R, Cohen W (2005). Extracting personal names from email. *Proc HLT/EMNLP*, Vancouver.

Mohit B, Hwa R (2005) Syntax-based semi-supervised named entity tagging. *Proc ACL*, Ann Arbor MI.

Sundheim B (1995), ed. *Proc MUC-6*, Columbia MD.

Sutton C, McCallum A (2005). Composition of CRFs for transfer learning, *Proc HLT/EMNLP*, Vancouver.

Kernel Regression Based Machine Translation

Zhuoran Wang and **John Shawe-Taylor**
Department of Computer Science
University College London
London, WC1E 6BT
United Kingdom
{z.wang, jst}@cs.ucl.ac.uk

Sandor Szedmak
School of Electronics and Computer Science
University of Southampton
Southampton, SO17 1BJ
United Kingdom
ss03v@ecs.soton.ac.uk

Abstract

We present a novel machine translation framework based on kernel regression techniques. In our model, the translation task is viewed as a string-to-string mapping, for which a regression type learning is employed with both the source and the target sentences embedded into their kernel induced feature spaces. We report the experiments on a French-English translation task showing encouraging results.

1 Introduction

Fig. 1 illustrates an example of phrase alignment for statistical machine translation (SMT). A rough linear relation is shown by the co-occurences of phrases in bilingual sentence pairs, which motivates us to introduce a novel study on the SMT task:

If we define the feature space $\mathcal{H}_x$ of our source language $\mathcal{X}$ as all its possible phrases (i.e. informative blended word *n*-grams), and define the mapping $\Phi_x : \mathcal{X} \to \mathcal{H}_x$, then a sentence $x \in \mathcal{X}$ can be expressed by its feature vector $\Phi_x(x) \in \mathcal{H}_x$. Each component of $\Phi_x(x)$ is indexed by a phrase with the value being the frequency of it in x. The definition of the feature space $\mathcal{H}_y$ of our target language $\mathcal{Y}$ can be made in a similar way, with corresponding mapping $\Phi_y : \mathcal{Y} \to \mathcal{H}_y$. Now in the machine translation task, given $S = \{(x_i, y_i) : x_i \in \mathcal{X}, y_i \in \mathcal{Y}, i = 1, \ldots, m\}$, a set of sample sentence pairs where y_i is the translation of x_i, we are trying to learn $\mathbf{W}$ a matrix represented linear operator, such that:

$$\Phi_y(y) = f(x) = \mathbf{W}\Phi_x(x) \quad (1)$$

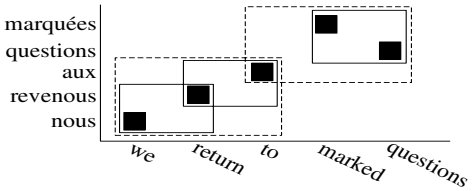


Figure 1: Phrase alignment in SMT

to predict the translation y for a new sentence x.

Comparing with traditional methods, this model gives us a theoretical framework to capture higher-dimensional dependencies within the sentences. To solve the multi-output regression problem, we investigate two models, least squares regression (LSR) similar to the technique presented in (Cortes et al., 2005), and maximum margin regression (MMR) introduced in (Szedmak et al., 2006).

The rest of the paper is organized as follows. Section 2 gives a brief review of the regression models. Section 3 details the solution to the pre-image problem. We report the experimental results in Section 4, with discussions in Section 5.

2 Kernel Regression with Vector Outputs

2.1 Kernel Induced Feature Space

In the practical learning process, only the inner products of the feature vectors are needed (see Section 2.2, 2.3 and 3), so we can perform the so-called kernel trick to avoid dealing with the very high-dimensional feature vectors explicitly. That is, for $x, z \in \mathcal{X}$, a kernel function is defined as:

$$\kappa_x(x, z) = \langle \Phi_x(x), \Phi_x(z) \rangle = \Phi_x(x)^\top \Phi_x(z) \quad (2)$$

Proceedings of NAACL HLT 2007, Companion Volume, pages 185–188,
Rochester, NY, April 2007.

Similarly, a kernel function $\kappa_y(\cdot,\cdot)$ is defined in $\mathcal{H}_y$.

In our case, the blended *n*-spectrum string kernel (Lodhi et al., 2002) that compares two strings by counting how many (contiguous) substrings of length from 1 up to *n* they have in common, is a good choice for the kernel function to induce our feature spaces $\mathcal{H}_x$ and $\mathcal{H}_y$ implicitly, even though it brings in some uninformative features (word n-grams) as well, when compared to our original definition.

2.2 Least Squares Regression

A basic method to solve the problem in Eq. 1 is least squares regression that seeks the matrix $\mathbf{W}$ minimizing the squared loss in $\mathcal{H}_y$ on the training set S:

$$\min \|\mathbf{W}\mathbf{M}_x - \mathbf{M}_y\|_F^2 \tag{3}$$

where $\mathbf{M}_x = [\Phi_x(x_1), ..., \Phi_x(x_m)]$, $\mathbf{M}_y = [\Phi_y(y_1), ..., \Phi_y(y_m)]$, and $\|\cdot\|_F$ denotes the Frobenius norm.

Differentiating the expression and setting it to zero gives:

$$\begin{aligned} & 2\mathbf{W}\mathbf{M}_x\mathbf{M}_x^\top - 2\mathbf{M}_y\mathbf{M}_x^\top = 0 \\ \Rightarrow \quad & \mathbf{W} = \mathbf{M}_y\mathbf{K}_x^{-1}\mathbf{M}_x^\top \end{aligned} \tag{4}$$

where $\mathbf{K}_x = \mathbf{M}_x^\top\mathbf{M}_x = (\kappa_x(x_i, x_j)_{1\le i,j\le m})$ is the Gram matrix.

2.3 Maximum Margin Regression

An alternative solution to our regression learning problem is proposed in (Szedmak et al., 2006), called maximum margin regression. If L2-normalized feature vectors are used in Eq. 1, denoted by $\bar{\Phi}_x(\cdot)$ and $\bar{\Phi}_y(\cdot)$, MMR solves the following optimization:

$$\begin{aligned} \min \quad & \frac{1}{2}\|\mathbf{W}\|_F^2 + C\sum_{i=1}^m \xi_i \\ \text{s.t.} \quad & \langle\bar{\Phi}_y(y_i), \mathbf{W}\bar{\Phi}_x(x_i)\rangle_{\mathcal{H}_y} \ge 1 - \xi_i, \\ & \xi_i > 0, i = 1, \ldots, m. \end{aligned} \tag{5}$$

where $C > 0$ is the regularization coefficient, and ξ_i are the slack variables. The Lagrange dual form with dual variables α_i gives:

$$\begin{aligned} \min \quad & \sum_{i,j=1}^m \alpha_i\alpha_j\bar{\kappa}_x(x_i, x_j)\bar{\kappa}_y(y_i, y_j) - \sum_{i=1}^m \alpha_i \\ \text{s.t.} \quad & 0 \le \alpha_i \le C,\ i = 1, \ldots, m. \end{aligned} \tag{6}$$

where $\bar{\kappa}_x(\cdot,\cdot)$ and $\bar{\kappa}_y(\cdot,\cdot)$ denote the kernel functions associated to the respective normalized feature vectors.

This dual problem can be solved efficiently with a perceptron algorithm based on an incremental subgradient method, of which the bounds on the complexity and achievable margin can be found in (Szedmak et al., 2006).

Then according to Karush-Kuhn-Tucker theory, $\mathbf{W}$ is expressed as:

$$\mathbf{W} = \sum_{i=1}^m \alpha_i\bar{\Phi}_y(y_i)\bar{\Phi}_x(x_i)^\top \tag{7}$$

In practice, MMR works better when the distribution of the training points are symmetrical. So we center the data before normalizing them. If $\Phi_{S_x} = \frac{1}{m}\sum_{i=1}^m \Phi_x(x_i)$ is the centre of mass of the source sentence sample set $\{x_i\}$ in the feature space, the new feature map is given by $\hat{\Phi}_x(\cdot) = \Phi_x(\cdot) - \Phi_{S_x}$. The similar operation is performed on $\Phi_y(\cdot)$ to obtain $\hat{\Phi}_y(\cdot)$. Then the L2-normalizations of $\hat{\Phi}_x(\cdot)$ and $\hat{\Phi}_y(\cdot)$ yield our final feature vectors $\bar{\Phi}_x(\cdot)$ and $\bar{\Phi}_y(\cdot)$.

3 Pre-image Solution

To find the pre-image sentence $y = f^{-1}(x)$ can be achieved by seeking y_t that has the minimum loss between its feature vector $\Phi_y(y_t)$ and our prediction $f(x)$. That is (Eq. 8: LSR, Eq. 9: MMR):

$$\begin{aligned} y_t &= \arg\min_{y\in\mathcal{Y}(x)} \|\mathbf{W}\Phi_x(x) - \Phi_y(y)\|^2 \\ &= \arg\min_{y\in\mathcal{Y}(x)} \kappa_y(y,y) - 2k_y(y)\mathbf{K}_x^{-1}k_x(x) \end{aligned} \tag{8}$$

$$\begin{aligned} y_t &= \arg\min_{y\in\mathcal{Y}(x)} 1 - \langle\bar{\Phi}_y(y), \mathbf{W}\bar{\Phi}_x(x)\rangle_{\mathcal{H}_y} \\ &= \arg\max_{y\in\mathcal{Y}(x)} \sum_{i=1}^m \alpha_i\bar{\kappa}_y(y_i, y)\bar{\kappa}_x(x_i, x) \end{aligned} \tag{9}$$

where $\mathcal{Y}(x) \subset \mathcal{Y}$ is a finite set covering all potential translations for the given source sentence x, and $k_x(\cdot) = (\kappa_x(\cdot, x_i)_{1\le i\le m})$ and $k_y(\cdot) = (\kappa_y(\cdot, y_i)_{1\le i\le m})$ are $m \times 1$ column matrices.

A proper $\mathcal{Y}(x)$ can be generated according to a lexicon that contains possible translations for every component (word or phrase) in x. But the size of it will grow exponentially with the length of x, which poses implementation problem for a decoding algorithm.

In earlier systems, several heuristic search methods were developed, of which a typical example is Koehn (2004)'s beam search decoder for phrase-based models. However, in our case, because of the $\kappa_y(y, y)$ item in Eq. 8 and the normalization operation in MMR, neither the expression in Eq. 8 nor the one in Eq. 9 can be decomposed into a sum of subfunctions each involving feature components in a local area only. It means we cannot estimate exactly how well a part of the source sentence is translated, until we obtain a translation for the entire sentence, which prevents us doing a straightforward beam search similar to (Koehn, 2004).

To simplify the situation, we restrict the reordering (distortion) of phrases that yield the output sentences by only allowing adjacent phrases to exchange their positions. (The discussion of this strategy can be found in (Tillmann, 2004).) We use $x_{[i:j]}$ and $y_{[i:j]}$ to denote the substrings of x and y that begin with the ith word and end with the jth. Now, if we go back to the implementation of a beam search, the current distortion restriction guarantees that in each expansion of the search states (hypotheses) we have $x_{[1:l_x]}$ translated to a $y_{[1:l_y]}$, either like state (a) or like state (b) in Fig. 2, where l_x is the number of words translated in the source sentence, and l_y is the number of words obtained in the translation.

We assume that if y is a good translation of x, then $y_{[1:l_y]}$ is a good translation of $x_{[1:l_x]}$ as well. So we can expect that the squared loss $\|\mathbf{W}\Phi_x(x_{[1:l_x]}) - \Phi_y(y_{[1:l_y]})\|^2$ in the LSR is small, or the inner product $\langle\bar{\Phi}_y(y_{[1:l_y]}), \mathbf{W}\bar{\Phi}_x(x_{[1:l_x]})\rangle_{\mathcal{H}_y}$ in the MMR is large, for the hypothesis yielding a good translation. According to Eq. 8 and Eq. 9, the hypotheses in the search stacks can thus be reranked with the following score functions (Eq. 10: LSR, Eq. 11: MMR):

$$Score(x_{[1:l_x]}, y_{[1:l_y]}) = \kappa_y(y_{[1:l_y]}, y_{[1:l_y]}) - 2k_y(y_{[1:l_y]})\mathbf{K}_x^{-1}k_x(x_{[1:l_x]}) \quad (10)$$

$$Score(x_{[1:l_x]}, y_{[1:l_y]}) = \sum_{i=1}^{m} \alpha_i \bar{\kappa}_y(y_i, y_{[1:l_y]})\bar{\kappa}_x(x_i, x_{[1:l_x]}) \quad (11)$$

Therefore, to solve the pre-image problem, we just employ the same beam search algorithm as (Koehn, 2004), except we limit the derivation of new hypotheses with the distortion restriction mentioned above. However, our score functions will bring more runtime complexities when compared with traditional probabilistic methods. The time complexity of a naive implementation of the blended *n*-spectrum string kernel between two sentences s_i and s_j is $O(n|s_i||s_j|)$, where $|\cdot|$ denotes the length of the sentence. So the score function in Eq. 11 results in an average runtime complexity of $O(mnl_yl)$, where l is the average length of the sentences y_i in the training set. Note here $\bar{\kappa}_x(x_{[1:l_x]}, x_i)$ can be pre-computed for l_x from 1 to $|x|$ before the beam search, which calls for $O(m|x|)$ space. The average runtime complexity of the score function in Eq. 10 will be the same if we pre-compute $\mathbf{K}_x^{-1}k_x(x_{[1:l_x]})$.

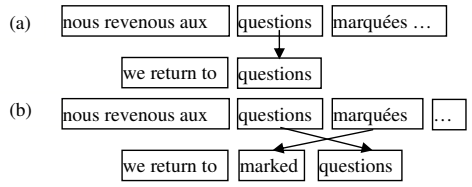


Figure 2: Search states with the limited distortion.

4 Experimental Results

4.1 Resource Description

Baseline System To compare with previous work, we take Pharaoh (Koehn, 2004) as a baseline system, with its default settings (translation table size 10, beam size 100). We train a trigram language model with the SRILM toolkit (Stocke, 2002). Whilst, the parameters for the maximum entropy model are developed based on the minimum error rate training method (Och, 2003).

In the following experiments, to facilitate comparison, each time we train our regression models and the language model and translation model for Pharaoh on a common corpus, and use the same phrase translation table as Pharaoh's to decode our systems. According to our preliminary experiments, with the beam size of 100, the search errors of our systems can be limited within 1.5%.

Corpora To evaluate our models, we randomly take 12,000 sentences from the French-English portion of the 1996–2003 Europarl corpus (Koehn, 2005) for scaling-up training, 300 for test (Test), and 300 for the development of Pharaoh (Dev). Some

	Vocabulary		Words		Perplexity	
	Fr	En	Fr	En	Dev	Test
4k	5084	4039	43k	39k	32.25	31.92
6k	6426	5058	64k	59k	30.81	29.03
8k	7377	5716	85k	79k	29.91	28.94
10k	8252	6339	106k	98k	27.55	27.09
12k	9006	6861	127k	118k	27.19	26.41

Table 1: Statistics of the corpora.

characteristics of the corpora are summarized in Table 1.

4.2 Results

Based on the 4k training corpus, we test the performance of the blended *n*-spectrum string kernel in LSR and MMR using BLEU score, with n increasing from 2 to 7. Fig. 3 shows the results. It can be found that the performance becomes stable when *n* reaches a certain value. Finally, we choose the 3-spectrum for LSR, and the 5-spectrum for MMR.

Then we scale up the training set, and compare the performance of our models with Pharaoh in Fig. 4. We can see that the LSR model performs almost as well as Pharaoh, whose differences of BLEU score are within 0.5% when the training set is larger than 6k. But MMR model performs worse than the baseline. With the training set of 12k, it is outperformed by Pharaoh by 3.5%.

5 Discussions

Although at this stage the main contribution is still conceptual, the capability of our approach to be applied to machine translation is still demonstrated. Comparable performance to previous work is achieved by the LSR model.

But a main problem we face is to scale-up the training set, as in practice the training set for SMT will be much larger than several thousand sentences. A method to speed up the training is proposed in (Cortes et al., 2005). By approximating the Gram matrix with a $n \times m$ $(n \ll m)$ low-rank matrix, the time complexity of the matrix inversion operation can be reduced from $O(m^3)$ to $O(n^2 m)$. But the space complexity of $O(nm)$ in their algorithm is still too expensive for SMT tasks. Subset selection techniques could give a solution to this problem, of

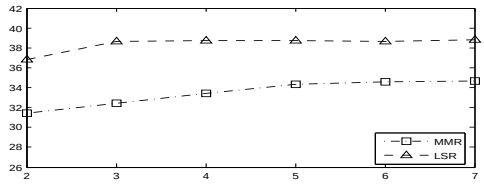


Figure 3: BLEU(%) versus *n*-spectrum

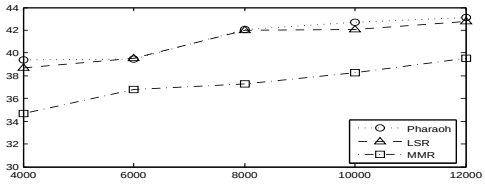


Figure 4: BLEU(%) versus training set size

which we will leave the further exploration to future work.

Acknowledgements

The authors acknowledge the support of the EU under the IST project No. FP6-033917.

References

C. Cortes, M. Mohri, and J. Weston. 2005. A general regression technique for learning transductions. In *Proc. of ICML'05*.

P. Koehn. 2004. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *Proc. of AMTA 2004*.

P. Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit X*.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. 2002. Text classification using string kernels. *J. Mach. Learn. Res.*, 2:419–444.

F. J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL'03*.

A. Stocke. 2002. SRILM – an extensible language modeling toolkit. In *Proc. of ICSLP'02*.

S. Szedmak, J. Shawe-Taylor, and E. Parado-Hernandez. 2006. Learning via linear operators: Maximum margin regression; multiclass and multiview learning at one-class complexity. Technical report, PASCAL, Southampton, UK.

C. Tillmann. 2004. A unigram orientation model for statistical machine translation. In *Proc. of HLT-NAACL'04*.

Modifying SO-PMI for Japanese Weblog Opinion Mining by Using a Balancing Factor and Detecting Neutral Expressions

Guangwei Wang
Graduate School of Information
Science and Technology
Hokkaido University
Sapporo, Japan 060-0814
wgw@media.eng.hokudai.ac.jp

Kenji Araki
Graduate School of Information
Science and Technology
Hokkaido University
Sapporo, Japan 060-0814
araki@media.eng.hokudai.ac.jp

Abstract

We propose a variation of the SO-PMI algorithm for Japanese, for use in Weblog Opinion Mining. SO-PMI is an unsupervised approach proposed by Turney that has been shown to work well for English. We first used the SO-PMI algorithm on Japanese in a way very similar to Turney's original idea. The result of this trial leaned heavily toward positive opinions. We then expanded the reference words to be sets of words, tried to introduce a balancing factor and to detect neutral expressions. After these modifications, we achieved a well-balanced result: both positive and negative accuracy exceeded 70%. This shows that our proposed approach not only adapted the SO-PMI for Japanese, but also modified it to analyze Japanese opinions more effectively.

1 Introduction

Recently, more and more websites add information in the form of personal opinions to the Web, e.g. customer reviews of products, forums, discussion groups, and blogs. Here, we use the term Weblog for these sites. This type of information is often useful. However, we have to deal with an enormous amount of unstructured and/or semi-structured data. These data are subjective, in free format and mostly textual, thus using them is difficult and time consuming. Therefore, how to mine the Weblog opinions automatically more effectively has attracted more and more attention (Gamon, 2005; Popescu, 2005; Chaovalit, 2005).

Turney (2002) has presented an unsupervised opinion classification algorithm called SO-PMI (Semantic Orientation Using Pointwise Mutual Information). The main use of SO-PMI is to estimate the semantic orientation (i.e. positive or negative) of a phrase by measuring the hits returned from a search engine of pairs of words or phrases, based on the mutual information theory. This approach has previously been successfully used on English. The average accuracy was 74% when evaluated on 410 reviews from Epinions[1].

However, according to our preliminary experiment, directly translating Turney's original idea into Japanese gave a very slanted result, with a *positive accuracy* of 95% and a *negative accuracy* of only 8%. We found that the balance between the positive and negative sides is influenced greatly by the page hits of reference words/sets, since a search engine is used. Therefore, we introduced a balancing factor according for the difference in occurrence between positive and negative words. And then we added several threshold rules to detect neutral expressions. The proposed approach is evaluated on 200 positive and 200 negative Japanese opinion sentences and yielded a well-balanced result.

In the remainder of this paper, we review the SO-PMI Algorithm in Section 2, then adapt the SO-PMI for Japanese and present the modifications in Section 3. In section 4, we evaluate and discuss the experimental results. Section 5 gives concluding remarks.

2 Details of the SO-PMI Algorithm

The SO-PMI algorithm (Turney, 2002) is used to estimate the semantic orientation (SO) of a phrase by

[1] http://www.epinions.com

Proceedings of NAACL HLT 2007, Companion Volume, pages 189–192,
Rochester, NY, April 2007.

measuring the similarity of pairs of words or phrases using the following formula:

$$PMI(word_1, word_2) = log_2\left[\frac{p(word_1 \& word_2)}{p(word_1)p(word_2)}\right] \quad (1)$$

$$SO(phrase) = PMI(phrase, \text{“}excellent\text{”}) - PMI(phrase, \text{“}poor\text{”}) \quad (2)$$

The reference words “excellent” and “poor” are used, thus SO is positive when a phrase is more strongly associated with “excellent” and negative when a phrase is more strongly associated with “poor”. Let hits(query) be the number of hits returned when using a search engine, the following estimate of SO can be derived from Formula (2) and (1) with some minor algebraic manipulation.

$$SO(phrase) = log_2[A]$$
$$A = \frac{hits(phrase\ NEAR\ \text{“}excellent\text{”}) * hits(\text{“}poor\text{”})}{hits(phrase\ NEAR\ \text{“}poor\text{”}) * hits(\text{“}excellent\text{”})} \quad (3)$$

Turney used AltaVista[2] search engine because it has a NEAR operator. This operator constrains the search to documents that contain the words within ten words of one another, in either order. Turney’s previous work has shown that NEAR performs better than AND when measuring the strength of semantic association between words.

3 Our Proposed Approach

The first step of our approach is to extract opinion phrases using word POS (part of speech) templates based on our analysis of opinions in Japanese Weblog and the results of related work (Kobayashi, 2003; Taku, 2002; Wang, 2006). The second step is to estimate the semantic orientation of the extracted phrases, using the SO-PMI algorithm.

3.1 Adapting SO-PMI for Japanese

Following Turney’s original idea, we first translated the SO formula to the one shown in Formula (4) for Japanese.

$$SO(phrase) = log_2[B] \quad (4)$$
$$B = \left[\frac{hits(phrase\ \ AND\ \ \text{"すばらしい"}) * hits(\text{"不良"})}{hits(phrase\ \ AND\ \ \text{"不良"}) * hits(\text{"すばらしい"})}\right]$$

We used the Google search engine[3] to get the hits(query) even though Google does not have a NEAR operator. The AltaVista NEAR operator does not work well for Japanese and Google indexes more pages than AltaVista, thus we used Google and replaced the NEAR operator with the AND operator in the SO formula. “すばらしい” and “不良” were selected because they correspond to the English words “excellent” and “poor”.

For testing the performance of this trial, we used 200 positive and 200 negative Japanese opinion sentences which have been labeled by hand. The results were very slanted. Many phrases, whether positive or negative in meaning, still received a positive SO. Some possible causes could be that “不良 (poor)” has more hits than “すばらしい (excellent)”, as shown in Table 1, and that the AND operator is less useful than the NEAR operator.

3.2 Modifying SO-PMI for Japanese

In Japanese, there are many expressions when people evaluate something. For example, “いい (good)”, “良い (good)”, “満足 (satisfaction)” , “すばらしい (excellent)” are usually used when someone wants to convey a positive opinion. Hence we tried to replace the reference words “excellent” and “poor” with two reference sets: “p_basic” and “n_basic”:

$$SO(phrase) = log_2[C]$$
$$C = \frac{hits(phrase\ AND p\_basic) * hits(n\_basic)}{hits(phrase\ AND n\_basic) * hits(p\_basic)} \quad (5)$$

“p_basic” is a set of common strong positive words in Japanese. “n_basic” is a set of common weak negative words. The hit counts of these words from Google is shown in Table 1 (All data from 2007/01/12). The hits(query) was calculated by *hits(phrase AND (“いい (good)” OR “好き (like)”) OR “良い (good)” OR ...)*.

Table 1: Frequency of p_basic/n_basic words on the Web

p_basic words	Hits (K)	R(%)	n_basic words	Hits (K)	R(%)
いい(good)	372,000	36.83	不良(poor)	119,000	11.78
好き(like)	242,000	23.96	悪い(bad)	110,000	10.89
良い(good)	211,000	20.89	不安(worry)	83,000	8.22
魅力(charm)	150,000	14.85	欠点(fault)	77,900	7.71
大好き(favorite)	115,000	11.39	難しい(hard)	77,600	7.68
欲しい(want)	115,000	11.39	嫌(dislike)	65,000	6.44
楽しい(delightful)	107,000	10.59	あんまり(not good)	37,900	3.75
よい(good)	103,000	10.20	嫌い(dislike)	37,100	3.67
良く(good)	96,900	9.59	だめ(useless)	26,500	2.62
満足(satisfaction)	80,600	7.98	辛い(painful)	26,500	2.62
面白い(interesting)	79,700	7.89	不快(dissatisfaction)	26,100	2.58
嬉しい(happy)	75,500	7.48	不満(dissatisfaction)	22,200	2.20
素敵(lovely)	74,700	7.40	最悪(worst)	20,700	2.05
うれしい(happy)	59,500	5.89	不具合 (fault)	16,600	1.64
おもしろい(interesting)	28,400	2.81	あまり(not good)	15,600	1.54
すばらしい(excellent)	26,000	2.57	まずい(bad)	10,300	1.02

We evaluated this modification using the same

[2] http://www.altavista.com/sites/search/adv

[3] http://www.google.co.jp/

data as in Section 3.1. We obtained a slightly better result. However the SO values were still slanted. This time many phrases, whether positive or negative in meaning, still received a negative SO. All of these test results are shown in detail in Section 4.2.

In the experiments above, we obtained heavily slanted results. We consider that the large difference in page hits between the positive and negative reference words/sets are the main cause for this phenomenon. To mitigate this problem, we decided to introduce a balancing factor to adjust the balance between the positive and negative sides. The SO formula was modified from (5) to (6).

$$SO(phrase) = log_2 [C] + f(\alpha) \quad (6)$$

The balancing factor $f(\alpha)$ was calculated by Formula (7).

$$f(\alpha) = \alpha * log_2 \left[\frac{hits(p\_basic)}{hits(n\_basic)} \right] \quad (7)$$

The log_2 of "p_basic" and "n_basic" is a factor that adjusts the balance of the similarity of "p_basic"/"n_basic" and phrases automatically by the hits of "p_basic"/"n_basic" itself. α is a weight value. We evaluated different values of α from "0.0" to "1.0" on the benchmark dataset, which is shown in detail in Section 4.2.

From these preliminary trials, we also found that many neutral phrases often receive positive or negative SO. Therefore we added detection of neutral expressions. The idea is that if the phrase is strongly or faintly associated with both "p_basic" and "n_basic", it is considered a neutral phrase. Because this means that this phrase has an ambiguous connection with both "p_basic" and "n_basic". We use the following rules (Figure 1) to separate neutral phrases from positive/negative phrases. The threshold values **ta, tb** and **tc** are obtained from a small, hand-labeled corpus.

*1. hits(phrase AND p_basic) > **ta** AND hits(phrase AND n_basic) > **ta***
*2. hits(phrase AND p_basic) < **tb** AND hits(phrase AND n_basic) < **tb***
*3. | hits(phrase AND p_basic) – hits(phrase AND n_basic) | < **tc***
4. SO(phrase) = 0

Figure 1: Rules for Detecting Neutral Expressions

4 Experimental Performance Evaluation

4.1 Gold Standard and Evaluation Metrics

As a gold standard, we collected a benchmark dataset which has 200 positive opinion sentences and 200 negative opinion sentences from the reviews about Electronic Dictionary and MP3 Player products that have been labeled as either positive or negative reviews in "Kakaku.com"[4]. "Kakaku.com" is the largest Japanese Weblog specializing in product comparison of consumer goods, including price and user opinions, etc. Lots of people exchange miscellaneous product information and reviews. These reviews are classified as questions, positive reviews, negative reviews, rumors, sale information or "other" category.

To classify a sentence as positive (P) or negative (N), the average SO of the phrases in the sentence is used. If the average SO is P, the sentence is a positive sentence; otherwise it is a negative sentence. As evaluation metrics, we measured our proposed approach's performance by *accuracy*. *accuracy* was measured as the number of sentences correctly classified as P/N sentences to the total number of P/N sentences in the benchmark dataset (200). **PA** means *positive accuracy*, **NA** means *negative accuracy*, i.e. the accuracy on only positive or negative sentences respectively.

4.2 Experiments and Results

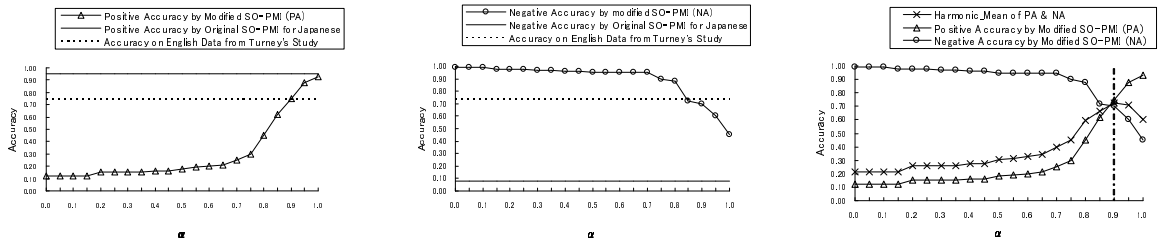
First we did the balancing factor experiment to determine the value of "α", using the benchmark dataset. The results are shown in Figure 2. (a) and (b) show the dashed line indicates average accuracy (74%) on English Data from Turney's Study (2002). Turney didn't evaluate positive and negative accuracy respectively. The full drawn line indicates the result after translating the original SO-PMI to Japanese (PA:95%, NA: 8%). **PA** series (the line with triangle mark)/**NA** series (the line with circle mark) when values of "α" from "0.0" to "1.0" were used.

Changing the α tends to be a tradeoff, lowering **PA** when **NA** is improved and vice versa. Therefore, we used $Harmonic\_Mean$ by the following formula to find a proper value of "α".

$$Harmonic\_Mean = \frac{2 * PA * NA}{PA + NA} \quad (8)$$

Figure 2, (c) shows **PA**, **NA** and $Harmonic\_Mean$ curves for different values

[4] http://www.kakaku.com/



(a) Positive Accuracy (PA) (b) Negative Accuracy (NA) (c) Harmonic-Mean of PA/NA

Figure 2: Experiment for α in Balance Factor

of "α". We selected the "α=0.9" giving the highest $Harmonic\_Mean$ value, thus giving a good balance between **PA** (75%) and **NA** (70%).

The comparative experiment results between the SO-PMI for Japanese (Test 1), and our modifications (Test 2, 3, 4) are shown in Table 2.

Table 2: Comparative Experiment Results

	Test Content	PA(%)	NA(%)
Test 1	Naive translation of Turney's Approach for Japanese	95	8
Test 2	Modification 1: Two Reference Sets	12	99
Test 3	Test 2 + Modification 2: Balancing Factor [α = 0.9]	75	70
Test 4	Test 3 + Modification 3: Neutral Phrase Detection	**78**	**72**

PA: Positive Accuracy NA: Negative Accuracy

In Test 1 and 2, we obtained extreme results, leaning to the positive or negative end, whether using the Turney's original approach or expanding the reference word as "p_basic" and "n_basic". In Test 3, we added a balancing factor as described in section 3.2, and obtained a comparatively well-balanced result. Finally, after adding the neutral expressions detection, we achieved a **PA** of 78% and **NA** of 72% (Test 4). The balance between positive and negative sides was quite improved by contrast with Test 1 and 2.

5 Conclusions

This study first proposed a modified unsupervised approach (SO-PMI) for Japanese Weblog Opinion Mining. Some parts of Turney's approach, such as the NEAR operator, does not work for Japanese, thus some modifications must be done. In a preliminary experiment, the *negative accuracy* (8%) was very poor while the *positive accuracy* (95%) was high. To deal with this phenomenon, we presented three modifications based on the characteristics of Japanese and the results of related work. The experiment results (*positive accuracy*: 78%, *negative accuracy*: 72%) show that our proposal achieved a considerably improved performance, comparing with directly translating the SO-PMI. Hence it would be expected that the balancing factor and neutral expressions detection would work effectively also for other reference words or languages. In the future, we will evaluate different choices of words for the sets of positive and negative reference words. We also plan to appraise our proposal on other languages.

References

Peter D. Turney. 2002. *Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews*. Proceedings 40th Annual Meeting of the ACL, pp. 417-424.

Popescu, Ana-Maria, and Oren Etzioni. 2005. *Extracting Product Features and Opinions from Reviews*. Proceedings of HLT-EMNLP.

Michael Gamon, Anthony Aue, Simon Corston-Oliver and Eric K. Ringger. 2005. *Pulse: Mining Customer Opinions from Free Text*. Proceedings of the 2005 Conference on Intelligent Data Analysis (IDA), pp.121-132.

Pimwadee Chaovalit and Lina Zhou. 2005. *Movie Review Mining: a Comparison between Supervised and Unsupervised Classification Approaches*. Proceedings of the 38th Annual HICSS.

Nozomi Kobayashi, Kentaro Inui, Yuji Matsumoto, Kenji Tateishi and Toshikazu Fukushima. 2003. *Collecting evaluative expressions by a text mining technique*. IPSJ SIG NOTE, Vol.154, No.12, In Japanese.

Taku Kudoh and Yuji Matsumoto. 2002. *Applying Cascaded Chunking to Japanese Dependency Structure Analysis*. Information Processing Society of Japan (IPSJ)Academic Journals, Vol 43, No 6, pp. 1834-1842, In Japanese.

Guangwei Wang and Kenji Araki. 2006. *A Decision Support System Using Text Mining Technology*. IEICE SIG Notes WI2-2006-6, pp. 55-56.

Combined Use of Speaker- and Tone-Normalized Pitch Reset with Pause Duration for Automatic Story Segmentation in Mandarin Broadcast News

Lei Xie, Chuan Liu and Helen Meng
Human-Computer Communications Laboratory
Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong, Hong Kong SAR of China
{lxie, cliu3, hmmeng}se.cuhk.edu.hk

Abstract

This paper investigates the combined use of pause duration and pitch reset for automatic story segmentation in Mandarin broadcast news. Analysis shows that story boundaries cannot be clearly discriminated from utterance boundaries by speaker-normalized pitch reset due to its large variations across different syllable tone pairs. Instead, speaker- and tone-normalized pitch reset can provide a clear separation between utterance and story boundaries. Experiments using decision trees for story boundary detection reinforce that raw and speaker-normalized pitch resets are not effective for Mandarin Chinese story segmentation. Speaker- and tone-normalized pitch reset is a good story boundary indicator. When it is combined with pause duration, a high *F*-measure of 86.7% is achieved. Analysis of the decision tree uncovered four major heuristics that show how speakers jointly utilize pause duration and pitch reset to separate speech into stories.

1 Introduction

Pitch reset refers to the speaker's general pitch declination through the course of a speech unit, followed by a reset to a high pitch at the start of next speech unit, as shown in Figure 1(a). The speech unit may be of different levels of granularity (Tseng et. al., 2005), such as a speech segment that conveys a central topic (e.g. a news story), a prosodic phrase group (PG) or an utterance. These units are often separated by pauses. Pauses and pitch resets were shown to be effective story boundary indicators in English broadcast news segmentation (Shriberg et. al., 2000; Tür et. al., 2001). These previous efforts specifically point out that pause durations are longer and pitch resets are more pronounced at story boundaries, when compared to utterance boundaries in English broadcast news. However, such story segmentation approaches may be different for a tonal language such as Mandarin Chinese. The use of similar prosodic features for Chinese news story segmentation deserves further investigation. The main reason is that Chinese tonal syllables may complicate the expressions of pitch resets. Chinese syllable tones are expressed acoustically in pitch trajectories, i.e, different tones show different pitch value ranges and trajectory patterns,[1] as shown in Figure 1(b). Initial work in (Levow, 2004) has shown that Mandarin words at story ending positions show a lower pitch as compared with words at non-story-ending positions. In this paper, we present a data-oriented study to investigate how the tonality of Mandarin syllables affects pitch resets at utterance and story boundaries. To alleviate the effects from tonality, we propose to use speaker- and tone-normalized pitch reset with pause duration to separate Mandarin broadcast audio stream into distinct news stories.

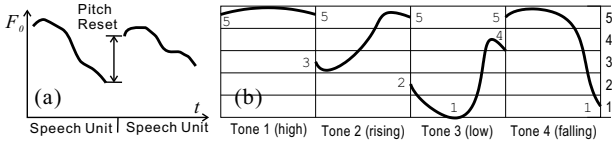


Figure 1: (a) Pitch reset phenomenon between speech units; (b) Pitch trajectories for the four Mandarin basic syllable tones. The speaker pitch range is segmented to five zones from high to low. The pitch trajectories of the four tones are 5-5, 3-5, 2-1-4 and 5-1, respectively.

2 Task and Corpus

In a continuous audio stream of broadcast news, there are programs that consist of speaker changes among anchors, reporters and interviewees. Other programs may contain a sequence of news stories reported by a single speaker. We focus on the latter kind in this investigation, because the combined use of pause duration and pitch reset to punctuate the end of a story and the beginning of the next carries many speaker-dependent characteristics.

We select a subset of TDT2 VOA Mandarin broadcast news corpus (LDC, 1998) and manually extract the news sessions reported by a single speaker. We also annotate

[1] http://www.mandarinbook.net/pronunciation/

Proceedings of NAACL HLT 2007, Companion Volume, pages 193–196,
Rochester, NY, April 2007.

Table 1: The TDT2 subset used in this study.

Nature	Mandarin news sessions reported by a single speaker (13.4 hours)
# of News Sessions	175 (Training: 74, Development: 50, Testing: 51)
Mean Session Duration	276 seconds, 1071 Mandarin characters
# of Story Boundaries	1085 (Training: 442, Development: 316, Testing: 327)
# of Speakers	11 (7 females and 4 males)
Mean Story Duration	36 seconds, 105 Mandarin characters
Transcriptions	Dragon ASR recognizer, GB-encoded word-level transcriptions in XML format

the news story boundaries in this subset. These single-speaker sessions typically contain between 3 to 9 short news stories separated by pauses and constitute about 30% of the entire TDT2 Mandarin corpus (by time duration). The selected subset is divided into training, development and testing sets. Details are shown in Table 1.

3 Region of Interest and Pitch Extraction

Previous work on English news segmentation (Shriberg et. al., 2000) measured pitch resets at inter-word boundaries. Since Chinese news transcripts come as a character stream and each character is pronounced as a tonal syllable, it is more reasonable to investigate the pitch reset phenomenon at the syllable level. We assume that a story boundary must occur at an utterance boundary. The utterances are separated by labeled pauses in the VOA transcriptions ([P] in Figure 2) and a story may contain various utterances (between 2 to 38 in the corpus). Therefore, we only investigate pitch resets in inter-syllable regions across two consecutive utterances as shown in Figure 2. This is reasonable because there are only 6 story boundaries (out of 1085) that are not signaled by pause breaks in the corpus. The region of interest (ROI) is limited to only two tonal syllables, i.e., the last tonal syllable of the previous utterance and the first tonal syllable of the following utterance. We have performed experiments on window length selection and results have shown a wider window does not bring a noticeable improvement.

Raw pitch values are extracted by the YIN pitch tracker (Cheveigné et. al., 2002). The output pitch trajectories are ranked as "good" and "best" by the pitch tracker. Pitch values for unvoiced and pause segments are assigned to be zero. We keep the "best" pitch trajectories for pitch reset measurements. We focus on pitch resets in the ROIs and thus obtain pitch contours for the left and right tonal syllables for each ROIs. However, the corpus transcription does not provide time annotations for those tonal syllables. Therefore, in the pitch trajectory of an audio stream, we search forwards and backwards on both sides of the pause segment for the nearest non-zero pitch measurement sequences. The two pitch sequences found are used as the pitch contours for the left and right tonal syllables of the ROI, respectively. This approximation is reasonable because a Mandarin tonal syllable usually exhibits a continuous pitch contour within its time duration.

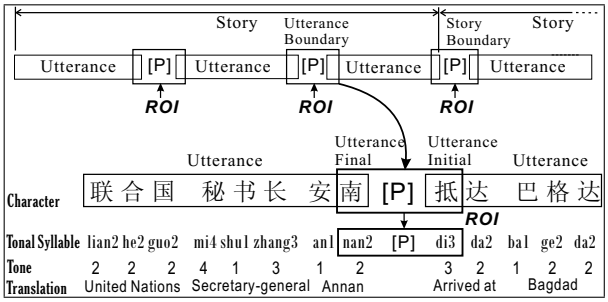


Figure 2: Region of interest(ROI) for pitch reset measure.

4 Speaker- and Tone-Normalized Pitch Reset Analysis in Mandarin Broadcast News

We investigate the pitch reset behavior in the ROIs, i.e., the pitch jump between the left and right tonal syllables at utterance and story boundaries across all corpus audio. Since pitch is a speaker-related feature, we adopt speaker-normalized pitch reset, defined as

$$PR = F_{0r} - F_{0l}, \quad (1)$$

where F_{0l} and F_{0r} are the speaker-normalized pitch for the left and right tonal syllables in the ROIs, which are calculated using

$$F_0 = (f_0 - \mu^s_{f_0})/\sigma^s_{f_0}. \quad (2)$$

f_0 denotes the mean value of the pitch contour of a tonal syllable uttered by speaker s. $\mu^s_{f_0}$ and $\sigma^s_{f_0}$ are the pitch mean and standard deviation calculated for speaker s over all the ROIs of speaker s in the corpus.

We measure the speaker-normalized pitch resets in all ROIs, and categorize them into two boundary types, i.e. utterance boundary and story boundary. To show the effects of tonality in pitch movement, we also categorize the pitch resets by different tone combinations (16 combinations for 4 Mandarin tones[2]). Figure 3 plots the mean PR of each tone combinations for the two boundary types calculated on the corpus data. We see that the pitch reset phenomenon holds for all tone combinations, even for the tone pair (1,3) (i.e. high, low) that has a very small reset. We perform *t*-tests ($p < 0.0025$, one-tailed), which show that for a given tone pair across a boundary, there is a significant difference in PR between an utterance boundary and a story boundary. However, the PR values vary greatly across different tone pairs. For example,

[2]The neutral tone is not considered here since its pitch pattern depends heavily on its neighboring tonal syllables.

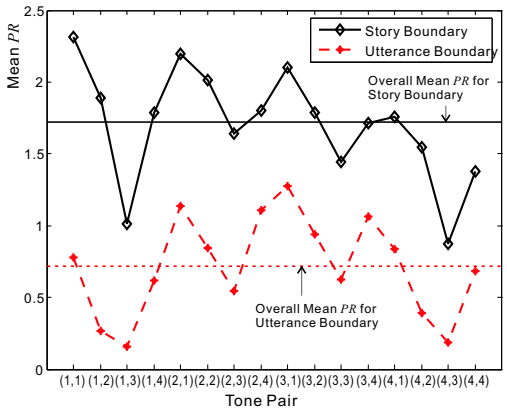


Figure 3: Mean speaker-normalized pitch reset of the 16 tone pairs for story and utterance boundaries.

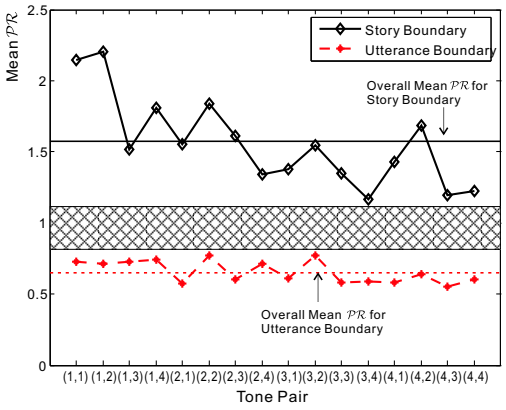


Figure 4: Mean speaker- and tone-normalized pitch reset of the 16 tone pairs for story and utterance boundaries.

pitch resets are reduced for the tone pairs (1,3) and (4,3), but are pronounced for the tone pairs (3,1) and (2,1). The *t*-test ($p < 0.0025$, one-tailed) shows that the PR difference between utterance boundaries and story boundaries are *not* significant. This motivates us to formulate a definition for speaker- and tone-normalized pitch reset.

The speaker- and tone-normalized pitch reset is defined as:

$$\mathcal{PR} = \mathcal{F}_{0r} - \mathcal{F}_{0l}, \quad (3)$$

where $\mathcal{F}_{0l}$ and $\mathcal{F}_{0r}$ are the speaker- & tone-normalized pitch for the left and right tonal syllables in the ROIs, respectively, defined as

$$\mathcal{F}_0 = (F_0 - \mu^{\tau}_{F_0})/\sigma^{\tau}_{F_0}, \quad (4)$$

where F_0 is the speaker-normalized pitch in Equation (2) of a tonal syllable with tone τ. $\mu^{\tau}_{F_0}$ and $\sigma^{\tau}_{F_0}$ are the pitch mean and standard deviation calculated for the tonal syllables with tone τ over all ROIs in the corpus. Figure 4 plots the mean $\mathcal{PR}$ of each tone combinations for the two boundary types calculated on the corpus data.

Figure 4 shows a clear separation in speaker- and tone-normalized pitch reset ($\mathcal{PR}$) between utterance and story boundaries (shade area in Figure 4). This result is statistically significant based on a *t*-test ($p < 0.0025$, one-tailed). This observation suggests that speaker- and tone-normalized pitch reset may be an effective story boundary indicator for Mandarin broadcast news.

5 Experiments on Story Boundary Detection

We perform experiments on story boundary detection at the ROIs in the corpus. Since all ROIs are utterance boundaries, of which only some are story boundaries, we take a "*hypothesize and classify*" approach in order to strike a good balance between recall and precision. We first hypothesize the occurrence of a story boundary if the ROI has a pause duration that exceeds a threshold. This is followed by a decision tree classifier that decides on the existence of a story boundary. We used Quinlan's C4.5-style decision tree (Quinlan, 1992) as the classifier, implemented by the IND toolkit.[3] The pause duration threshold was selected by a heuristic search procedure described as follows: We experimented with pause durations ranging from 0.1 to 4 seconds with step size of 0.1 second. In each case, we hypothesized raw boundaries in the training and development sets. A decision tree was then grown using the raw boundary hypotheses of the training set, and tested on the raw boundary hypotheses of the development set. The pause duration leading to the highest F-measure on the development set was selected as the optimal threshold for the further experiments on the testing set.

We develop seven story boundary detectors according to the features used (see Table 2). The boundary detection results on the testing set are shown in Table 2. From Table 2, we can see that the detector using pause duration achieves a high F-measure of 82.2%. This result is reasonable since VOA Mandarin news broadcast makes large use of long pauses at story boundaries, especially at news sessions reported by a single speaker. The detector using raw pitch reset ($pr = f_{0r} - f_{0l}$) only gets a F-measure of 50.8% and the speaker-normalized pitch reset (PR) achieves a slightly better F-measure of 55.3%. Speaker- and tone-normalized pitch reset ($\mathcal{PR}$) achieves a superior performance with an F-measure of 71.1%. This result is consistent with the observations in Section 4. The story boundary indicative ability of speaker-normalized pitch reset is affected by the tonality of Mandarin syllable. Speaker- and tone-normalized pitch reset can alleviate the effects, thus leading to a better discrimination. Based on Table 2, when pause is combined with raw pitch reset, the F-measure degrades from 82.2% to 68.3%. The F-measure reaches 77.4% when we combine pause with speaker-normalized pitch reset. When pause is combined with speaker- and tone-normalized pitch reset (*Pause*+$\mathcal{PR}$), the best F-measure is achieved at 86.7%.

[3] http://ic.arc.nasa.gov/projects/bayes-group/ind/

Table 2: Story boundary detection experiment results(%)

Feature	Recall	Precision	F-Measure
Pause	77.1	88.1	82.2
pr	52.0	49.7	50.8
PR	56.6	54.1	55.3
$\mathcal{PR}$	70.3	72.0	71.1
Pause+pr	66.4	70.3	68.3
Pause+PR	72.2	83.5	77.4
Pause+$\mathcal{PR}$	82.6	91.3	86.7

Table 3: Heuristics for story boundary decision

No.	Description	Story Boundary?
1	Pause duration is short ($P < 1.475$) and pitch reset is small ($\mathcal{PR} < 0.401$)	No
2	Pause duration is short ($P < 1.475$) and pitch reset is huge ($\mathcal{PR} > 1.112$)	Yes
3	Pause duration is long ($2.315 \leq P < 4.915$) and pitch reset is big ($\mathcal{PR} > 0.715$)	Yes
4	Pause duration is long ($P \geq 4.915$) and pitch reset is low ($\mathcal{PR} < 0.3513$)	No

Figure 5 shows the top levels of the decision tree obtained using the *Pause*+$\mathcal{PR}$ set. We can observe the complementarity between pause duration and pitch reset in story boundary detection. This may be summarized in terms of four major *heuristics* shown on the tree (labeled as 1 to 4 in Figure 5). These heuristics cover about 83% decisions made on the testing set, as described in Table 3.

Heuristics 2 is mainly used to detect possibly missing story boundaries with short pauses caused by speaker speaking style, e.g., reporters Li Weiqing and Yang Chen tend to use short pauses to separate news stories, but they tend to offset the reduced pauses with pronounced pitch resets to signify story boundaries. Heuristics 4 detects possibly false alarms due to broadcast interruptions in boundary detection. These interruptions (i.e. silences) usually occur within a news story and may last for several seconds (usually > 5 $seconds$).

6 Summary and Future Work

This paper investigated the combined use of pause duration and pitch reset for automatic story segmentation in Mandarin broadcast news. Pitch reset analysis on Mandarin broadcast news shows that story boundaries cannot be discriminated from utterance boundaries by speaker-normalized pitch reset, because speaker-normalized pitch reset varies greatly across different tone pairs of boundary syllables. This motivates us to investigate the speaker- and tone-normalized pitch reset. Analysis shows that speaker- and tone-normalized pitch reset can clearly separate utterance boundaries from story boundaries across all tone pairs. This result shows the difference between English and Chinese. Previous work for English (Shriberg et. al., 2000; Tür et. al., 2001) shows that speaker-normalized pitch reset is effective. This work shows that the same measurement is not sufficient for Chinese; instead we need to use speaker- and tone-normalized pitch reset in Chinese story segmentation. When pause duration is combined with speaker- and tone-normalized pitch reset, the best performance is achieved with a high F-measure of 86.7%. Analysis of the decision tree uncovered four major heuristics that show how speakers jointly utilize pause and pitch reset to separate speech into stories.

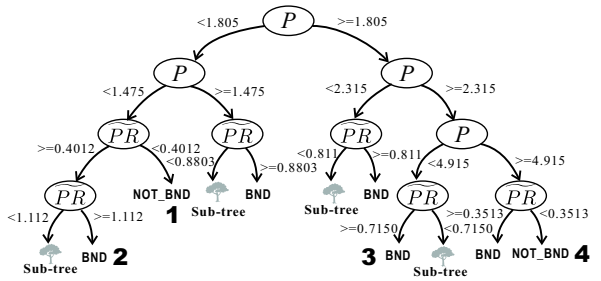


Figure 5: Decision tree for story boundary classification based on the *Pause*+$\mathcal{PR}$ feature set. BND denotes story boundary, and NOT_BND denotes not story boundary.

Future work will investigate the pitch reset phenomenon in Cantonese broadcast news, because Cantonese is another major Chinese dialect with more complicated tonal characteristics. We also plan to incorporate prosodic cues with lexical cues to further improve performance in Chinese story segmentation.

References

Shriberg E., Stolcke A., Hakkani-Tür D. and Tür G. 2000. Prosody-based automatic segmentation of speech into sentences and topics. *Speech Comm.*, 32(1-2):127–154.

Tür G. and Hakkani-Tür D. 2001. Integrating Prosodic and Lexical Cues for Automatic Topic Segmentation. *Computational Linguistics*, 27(1):31–57.

Levow G. A. 2004. Prosody-based Topic Segmentation for Mandarin Broadcast News. *Proc. of HLT-NAACL*, 137–140.

The Linguistic Data Consortium. 1998. *http://projects.ldc.upenn.edu/TDT2/.*

de Cheveigné A. and Kawahara H. 2002. Yin, a fundamental frequency estimator for speech and music. *Journal of the Acoustic Society of America*, 111(4):1917–1930.

Tseng C. Y., Pin S. H., Lee Y., Wang H. M. and Chen Y. C. 2005. Fluent speech prosody: Framework and modeling. *Speech Comm.*, 46:284–309.

Quinlan J. R. 1992. *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Chinese Named Entity Recognition with Cascaded Hybrid Model

Xiaofeng YU

Information Systems Laboratory
Department of Systems Engineering & Engineering Management
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
xfyu@se.cuhk.edu.hk

Abstract

We propose a high-performance cascaded hybrid model for Chinese NER. Firstly, we use Boosting, a standard and theoretically well-founded machine learning method to combine a set of weak classifiers together into a base system. Secondly, we introduce various types of heuristic human knowledge into Markov Logic Networks (MLNs), an effective combination of first-order logic and probabilistic graphical models to validate Boosting NER hypotheses. Experimental results show that the cascaded hybrid model significantly outperforms the state-of-the-art Boosting model.

1 Introduction

Named entity recognition (NER) involves the identification and classification of certain proper nouns in text, such as person names (PERs), locations (LOCs), organizations (ORGs), miscellaneous names (MISCs), temporal, numerical and monetary phrases. It is a well-established task in the NLP community and is regarded as crucial technology for many NLP applications, such as information extraction, question answering, information retrieval and machine translation.

Compared to European-language NER, Chinese NER seems to be more difficult (Yu *et al.*, 2006). Recent approaches to Chinese NER are a shift away from manually constructed rules or finite state patterns towards machine learning or statistical methods. However, rule-based NER systems lack robustness and portability, and machine learning approaches might be unsatisfactory to learn linguistic information in Chinese NEs. In fact, Chinese NEs have distinct linguistic characteristics in their composition and human beings usually use prior knowledge to recognize NEs. For example, about 365 of the highest frequently used surnames cover 99% Chinese surnames (Sun *et al.*, 1995). For the LOC "北京市/Beijing City", "北京/Beijing" is the name part and "市/City" is the salient word. For the ORG "北京市政府/Beijing City Government", "北京/Beijing" is the LOC name part, "市/City" is the LOC salient word and "政府/Government" is the ORG salient word. Some ORGs contain one or more PERs, LOCs, MISCs and ORGs. A more complex example is the nested ORG "湖北省武汉市武汉大学计算机学院/School of Computer Science, Wuhan University, Wuhan City, Hubei Province" which contains two ORGs "武汉大学/Wuhan University" and "计算机学院/School of Computer Science" and two LOCs "湖北省/Hubei Province" and "武汉市/Wuhan City". The two ORGs contain ORG salient words "大学/University" and "学院/School", while the two LOCs contain LOC salient words "省/Province" and "市/City" respectively.

Inspired by the above observation, we propose a cascaded hybrid model for Chinese NER [1]. First, we employ Boosting, which has theoretical justification and has been shown to perform well on other NLP problems, to combine weak classifiers into a strong classifier. We then exploit a variety of heuristic human knowledge into MLNs, a powerful combination of logic and probability, to validate Boosting NER hypotheses. We also use three Markov chain Monte Carlo (MCMC) algorithms for probabilistic inference in MLNs. Experimental results show that the cascaded hybrid model yields better NER results than the stand-alone Boosting model by a large margin.

2 Boosting

The main idea behind the Boosting algorithm is that a set of many simple and moderately accurate weak classifiers (also called *weak hypotheses*) can be effectively combined to yield a single strong classifier (also called the *final hypothesis*). The algorithm works by training weak classifiers sequentially whose classification accuracy is slightly better than random guessing and finally combin-

[1] In this paper we only focus on PERs, LOCs, ORGs and MISCs. Since temporal, numerical and monetary phrases can be well identified with rule-based approaches.

Proceedings of NAACL HLT 2007, Companion Volume, pages 197–200,
Rochester, NY, April 2007.

ing them into a highly accurate classifier. Each weak classifier searches for the hypothesis in the hypotheses space that can best classify the current set of training examples. Based on the evaluation of each iteration, the algorithm re-weights the training examples, forcing the newly generated weak classifier to give higher weights to the examples that are misclassified in the previous iteration.

We use the AdaBoost.MH algorithm (Schapire and Singer, 1999) as shown in Figure 1, an n-ary classification variant of the original well-known binary AdaBoost algorithm (Freund and Schapire, 1997). It has been demonstrated that Boosting can be used to build language-independent NER models that perform exceptionally well (Wu *et al.* (2002), Wu *et al.* (2004), Carreras *et al.* (2002)). In particular, reasonable Chinese NER results were still obtained using Boosting, even though there was no Chinese-specific tuning and the model was only trained on one-third of the provided corpora in SIGHAN bakeoff-3 (Yu *et al.*, 2006).

Input: A training set $T_r = \{< d_1, C_1 >, \ldots, < d_g, C_g >\}$ where $C_j \subseteq C = \{c_1, ..., c_m\}$ for all $j = 1, \ldots, g$.
Output: A final hypothesis $\Phi(d, c) = \sum_{s=1}^{S} \alpha_s \Phi_s(d, c)$.
Algorithm: Let$D_1(d_j, c_i) = \frac{1}{mg}$ for all $j = 1, \ldots, g$ and for all $i = 1, \ldots, m$. For $s = 1, \ldots, S$ do:

- pass distribution $D_s(d_j, c_i)$to the weak classifier;
- derive the weak hypothesis Φ_s from the weak classifier;
- choose $\alpha_s \in R$;
- set $D_{s+1}(d_j, c_i) = \frac{D_s(d_j,c_i)\exp(-\alpha_s C_j[c_i]\Phi_s(d_j,c_i))}{Z_s}$ where $Z_s = \sum_{i=1}^{m}\sum_{j=1}^{g} D_s(d_j, c_i)\exp(-\alpha_s C_j[c_i]\Phi_s(d_j, c_i))$ is a normalization factor chosen so that $\sum_{i=1}^{m}\sum_{j=1}^{g} D_{s+1}(d_j, c_i) = 1$.

Figure 1: The AdaBoost.MH algorithm.

3 Markov Logic Networks

A Markov Network (also known as Markov Random Field) is a model for the joint distribution of a set of variables (Pearl, 1988). It is composed of an undirected graph and a set of potential functions. A First-Order Knowledge Base (KB) (Genesereth and Nislsson, 1987) is a set of sentences or formulas in first-order logic. A Markov Logic Network (MLN) (Richardson and Domingos, 2006) is a KB with a weight attached to each formula (or clause). Together with a set of constants representing objects in the domain, it species a ground Markov Network containing one feature for each possible grounding of a first-order formula in the KB, with the corresponding weight. The weights associated with the formulas in an MLN jointly determine the probabilities of those formulas (and vice versa) via a *log-linear model*. An MLN defines a probability distribution over Herbrand interpretations (possible worlds), and can be thought of as a *template* for constructing Markov Networks. The probability distribution over possible worlds x specified by the ground Markov Network $M_{L,C}$ is given by

$$P(X = x) = \frac{1}{Z}\exp(\sum w_i n_i(x)) = \frac{1}{Z}\prod \phi_i\left(x_{\{i\}}\right)^{n_i(x)} \quad (1)$$

where F_i is the formula in first-order logic, w_i is a real number, $n_i(x)$ is the number of true groundings of F_i in x, $x_{\{i\}}$ is the true value of the atoms appearing in F_i, and $\phi_i\left(x_{\{i\}}\right) = e^{w_i}$.

3.1 Learning Weights

Given a relational database, MLN weights can in principle be learned generatively by maximizing the likelihood of this database. The gradient of the log-likelihood with respect to the weights is

$$\frac{\partial}{\partial w_i}\log P_w(X = x) = n_i(x) - \sum P_w(X = x')n_i(x') \quad (2)$$

where the sum is over all possible databases x', and $P_w(X = x')$ is $P(X = x')$ computed using the current weight vector $w = (w_1, ..., w_i, ...)$. Unfortunately, computing these expectations can be very expensive. Instead, we can maximize the *pseudo-likelihood* of the data more efficiently. If x is a possible database and x_l is the lth ground atom's truth value, the *pseudo-log-likelihood* of x given weights w is

$$\log P_w^*(X = x) = \sum_{l=1}^{n} \log P_w(X_{l=x_l} \mid MB_x(X_l)) \quad (3)$$

where $MB_x(X_l)$ is the state of X_l's Markov blanket in the data. Computing Equation 3 and its gradient does not require inference over the model, and is therefore much faster. We optimize the *pseudo-log-likelihood* using the limited-memory BFGS algorithm (Liu and Nocedal, 1989).

3.2 Inference

If F_1 and F_2 are two formulas in first-order logic, C is a finite set of constants including any constants that appear in F_1 or F_2, and L is an MLN, then

$$P\ (F_1 \mid F_2, L, C) = P(F_1 \mid F_2, M_{L,C}) \quad (4)$$

$$= \frac{P(F_1 \wedge F_2 \mid M_{L,C})}{P(F_2 \mid M_{L,C})} \quad (5)$$

$$= \frac{\sum_{x \in \chi_{F_1} \cap \chi_{F_2}} P(X = x \mid M_{L,C})}{\sum_{x \in \chi_{F_2}} P(X = x \mid M_{L,C})} \quad (6)$$

where χ_{F_i} is the set of worlds where F_i holds, and $P(x \mid M_{L,C})$ is given by Equation 1. The question of whether a knowledge base entails a formula F in first-order logic is the question of whether $P(F \mid L_{\mathrm{KB}}, C_{\mathrm{KB},F}) = 1$, where L_{KB} is the MLN obtained by assigning infinite weight to all the formulas in KB, and $C_{\mathrm{KB},F}$ is the set of all constants appearing in KB or F.

The most widely used approximate solution to probabilistic inference in MLNs is Markov chain Monte Carlo (MCMC) (Gilks *et al.*, 1996). In this framework, the Gibbs sampling algorithm is to generate an *instance* from the distribution of each variable in turn, conditional on the current values of the other variables. One way to speed up Gibbs sampling is by Simulated Tempering (Marinari and Parisi, 1992), which performs simulation in a *generalized ensemble*, and can rapidly achieve an equilibrium state. Poon and Domingos (2006) proposed MC-SAT, an inference algorithm that combines ideas from MCMC and satisfiability.

4 Heuristic Human Knowledge

Even though the Boosting model is able to accommodate a large number of features, some NEs, especially LOCs, ORGs and MISCs are difficult to identify due to lack of linguistic knowledge. For example, some ORGs are possibly mistagged as LOCs and/or MISCs. We incorporate heuristic human knowledge via MLNs to validate the Boosting NER hypotheses. We extract 151 location salient words and 783 organization salient words from the LDC Chinese-English bi-directional NE lists compiled from Xinhua News database. We also make a punctuation list which contains 19 items. We make the following assumptions to validate the Boosting results:

- Obviously, if a tagged entity ends with a location salient word, it is a LOC. If a tagged entity ends with an organization salient word, it is an ORG.
- If a tagged entity is close to a subsequent location salient word, probably they should be combined together as a LOC. The closer they are, the more likely that they should be combined.
- Heuristically, if a series of consecutive tagged entities are close to a subsequent organization salient word, they should probably be combined together as an ORG because an ORG may contain multiple PERs, LOCs, MISCs and ORGs.
- Similarly, if there exists a series of consecutive tagged entities and the last one is tagged as an ORG, it is likely that all of them should be combined as an ORG.
- Entity length restriction: all kinds of tagged entities cannot exceed 25 Chinese characters.
- Punctuation restriction: in general, all tagged entities cannot span any punctuation.
- Since all NEs are proper nouns, the tagged entities should end with noun words.

All the above human knowledge can be formulized as first-order logic to construct the structure of MLNs. And all the validated Boosting results are accepted as new NE candidates (or common nouns). We train an MLN to recognize them.

5 Experiments

We randomly selected 15,000 and 3,000 sentences from the People's Daily corpus as training and test sets, respectively. We used the *decision stump*[2] as the weak classifier in Boosting to construct a character-based Chinese NER baseline system.

The features used were as follows:

- The current character and its POS tag.
- The characters within a window of 3 characters before and after the current character.
- The POS tags within a window of 3 characters before and after the current character.
- A small set of conjunctions of POS tags and characters within a window of 3 characters of the current character.
- The BIO [3] chunk tags of the previous 3 characters.

We declared 10 *predicates* and specified 9 first-order formulas according to the heuristic human knowledge in Section 4. For example, we used `person(candidate)` to predicate whether a candidate is a PER. *Formulas* are recursively constructed from atomic formulas using logical connectives and quantifiers. They are constructed using four types of symbols: *constants*, *variables*, *functions*, and *predicates*. *Constant* symbols represent objects in the domain of interest (e.g., "北京/Beijing" and "上海/Shanghai" are LOCs). *Variable* symbols range over the objects in the domain. *Function* symbols represent mappings from tuples of objects to objects. *Predicate* symbols represent relations among objects in the domain or attributes of objects. For example, the formula `endwith(r, p)^locsalientword(p)=>location(r)` means if `r` ends with a location salient word `p`, then it is a LOC.

[2] A *decision stump* is basically a one-level decision tree where the split at the root level is based on a specific attribute/value pair.

[3] In this representation, each character is tagged as either the beginning of a named entity (B tag), a character inside a named entity (I tag), or a character outside a named entity (O tag).

We extracted all the distinct NEs (4,475 PERs, 2,170 LOCs, 2,823 ORGs and 614 MISCs) from the 15,000-sentence training corpus. An MLN training database, which consists of 10 *predicates* and 44,810 ground atoms was built. A *ground atom* is an atomic formula all of whose arguments are ground terms (terms containing no variables). For example, the ground atom `location(`北 京 市`)` conveys that "北 京 市/Beijing City" is a LOC.

During MLN learning, each formula is converted to Conjunctive Normal Form (CNF), and a weight is learned for each of its clauses. The weight of a clause is used as the mean of a Gaussian prior for the learned weight. These weights reflect how often the clauses are actually observed in the training data.

We validated 352 Boosting results to construct the MLN testing database, which contains 1,285 entries and these entries are used as *evidence* for inference. Inference is performed by grounding the minimal subset of the network required for answering the query predicates. We applied 3 MCMC algorithms: Gibbs sampling (GS), MC-SAT and Simulated Tempering (ST) for inference and the comparative NER results are shown in Table 1. The cascaded hybrid model greatly outperforms the Boosting model. We obtained the same results using GS and ST algorithms. And GS (or ST) yields slightly better results than the MC-SAT algorithm.

Table 1: Comparative Chinese NER Results

	Precision	Recall	$F_{\beta=1}$
Boosting			
PER	99.39%	99.06%	99.22
LOC	87.55%	91.81%	89.63
ORG	82.15%	66.61%	73.57
MISC	80.00%	87.84%	83.74
Overall	90.26%	89.42%	89.84
Hybrid (MC-SAT)			
PER	99.39%	99.06%	99.22
LOC	94.83%	91.81%	93.30
ORG	87.82%	85.69%	86.74
MISC	93.53%	85.10%	89.12
Overall	95.01%	92.78%	93.88
Hybrid (GS/ST)			
PER	99.39%	99.06%	99.22
LOC	94.80%	91.91%	93.34
ORG	87.82%	86.28%	87.04
MISC	93.53%	85.10%	89.12
Overall	94.99%	92.93%	93.95

6 Conclusion

In this paper we propose a cascaded hybrid model for Chinese NER. We incorporate human heuristics via MLNs, which produce a set of weighted first-order clauses to validate Boosting NER hypotheses. To the best of our knowledge, this is the first attempt at using MLNs for the NER problem in the NLP community. Experiments on People's Daily corpus illustrate the promise of our approach. Directions for future work include learning the structure of MLNs automatically and using MLNs for information extraction and statistical relational learning (e.g., entity relation identification).

References

Xavier Carreras, Lluís Màrquez, and Lluís Padró. Named entity extraction using AdaBoost. In *Computational Natural Language Learning (CoNLL-2002), at COLING-2002*, pages 171–174, Taipei, Sep 2002.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Computer and System Sciences*, 55(1):119–139, 1997.

Michael R. Genesereth and Nils J. Nislsson. *Logical foundations of artificial intelligence*. Morgan Kaufmann Publishers Inc., San Mateo, CA, 1987.

W.R. Gilks, S. Richardson, and D.J. Spiegelhalter. *Markov chain Monte Carlo in practice*. Chapman and Hall, London, UK, 1996.

Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

Enzo Marinari and Giorgio Parisi. Simulated Tempering: A new Monte Carlo scheme. *Europhysics Letters*, 19:451–458, 1992.

Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, 1988.

Hoifung Poon and Pedro Domingos. Sound and efficient inference with probabilistic and deterministic dependencies. In *21st National Conference on Artificial Intelligence (AAAI-06)*, Boston, Massachusetts, July 2006. The AAAI Press.

Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.

Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

Maosong Sun, Changning Huang, Haiyan Gao, and Jie Fang. Identifying Chinese names in unrestricted texts. *Journal of Chinese Information Processing*, 1995.

Dekai Wu, Grace Ngai, Marine Carpuat, Jeppe Larsen, and Yongsheng Yang. Boosting for named entity recognition. In *Computational Natural Language Learning (CoNLL-2002), at COLING-2002*, pages 195–198, Taipei, Sep 2002.

Dekai Wu, Grace Ngai, and Marine Carpuat. Why nitpicking works: Evidence for Occam's razor in error correctors. In *20th International Conference on Computational Linguistics (COLING-2004)*, Geneva, 2004.

Xiaofeng Yu, Marine Carpuat, and Dekai Wu. Boosting for Chinese named entity recognition. In *5th SIGHAN Workshop on Chinese Language Processing*, Australia, July 2006.

A Three-step Deterministic Parser for Chinese Dependency Parsing

Kun Yu
Graduate School of Informatics
Kyoto University
kunyu@nlp.kuee.kyoto-u.ac.jp

Sadao Kurohashi
Graduate School of Informatics
Kyoto University
kuro@i.kyoto-u.ac.jp

Hao Liu
Graduate School of Information Science and Technology
The University of Tokyo
liuhao@kc.t.u-tokyo.ac.jp

Abstract

This paper presents a three-step dependency parser to parse Chinese deterministically. By dividing a sentence into several parts and parsing them separately, it aims to reduce the error propagation coming from the greedy characteristic of deterministic parsing. Experimental results showed that compared with the deterministic parser which parsed a sentence in sequence, the proposed parser achieved extremely significant improvement on dependency accuracy.

1 Introduction

Recently, as an attractive alternative to probabilistic parsing, deterministic parsing (Yamada and Matsumoto, 2003; Nivre and Scholz, 2004) has drawn great attention with its high efficiency, simplicity and good accuracy comparable to the state-of-the-art generative probabilistic models. The basic idea of deterministic parsing is using a greedy parsing algorithm that approximates a globally optimal solution by making a sequence of locally optimal choices (Hall et al., 2006). This greedy idea guarantees the simplicity and efficiency, but at the same time it also suffers from the error propagation from the previous parsing choices to the left decisions.

For example, given a Chinese sentence, which means *Paternity test is a test that gets personal identity through DNA analysis, and it brings proof for finding lost children*, the correct dependency tree is shown by solid line (see Figure 1). But, if word 通过(through) is incorrectly parsed as depending on word 是(is) (shown by dotted line), this error will result in the incorrect parse of word 鉴定(a test) as depending on word 提供 (brings) (shown by dotted line).

This problem exists not only in Chinese, but also in other languages. Some efforts have been done to solve this problem. Cheng et al. (2005) used a root finder to divide one sentence into two parts by the root word and parsed them separately. But the two-part division is not enough when a sentence is composed of several coordinating sub-sentences. Chang et al. (2006) applied a pipeline framework in their dependency parser to make the local predictions more robust. While it did not show great help for stopping the error propagation between different parsing stages.

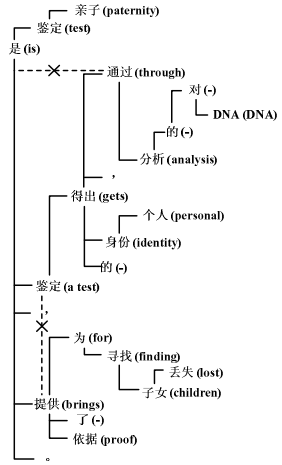


Figure 1. Dependency tree of a sentence (word sequence is top-down)

This paper focuses on resolving this issue for Chinese. After analyzing the dependency structure of sentences in Penn Chinese Treebank 5.1 (Xue et al., 2002), we found an interesting phenomenon: if we define a **main-root** as *the head of a sentence*, and define a **sub-sentence** as *a sequence of words separated by punctuations, and the head*[1] *of these words is the child of main-root or main-root itself*, then the punctuations that depend on main-root can be a separator of sub-sentences.

For example, in the example sentence there are three punctuations marked as PU_A, PU_B and PU_C, in which PU_B and PU_C depends on main-root but PU_A depends on word 得出(gets). According to our observation, PU_B and PU_C can be used for segmenting this sentence into two sub-sentences A and B (circled by dotted line in Figure 2), where the sub-root of A is main-root and the sub-root of B depends on main-root.

This phenomenon gives us a useful clue: *if we divide a sentence by the punctuations whose head is main-root, then the divided sub-sentences are basically independent of each other, which means we can parse them separately*. The shortening of sentence length and the recognition of sentence structure guarantee the robustness of deterministic parsing. The independent parsing of each sub-sentence also prevents the error-propagation. In

[1] *The head of sub-sentence* is defined as a **sub-root**.

Proceedings of NAACL HLT 2007, Companion Volume, pages 201–204,
Rochester, NY, April 2007.

addition, because the sub-root depends on main-root or is main-root itself, it is easy to combine the dependency structure of each sub-sentence to create the final dependency tree.

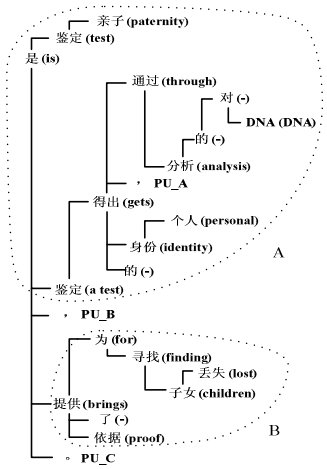


Figure 2. A segmentation of the sentence in Figure 1

Based on above analyses, this paper proposes a three-step deterministic dependency parser for Chinese, which works as:

Step1(Sentence Segmentation): Segmenting a sentence into sub-sentences by punctuations (sub-sentences do not contain the punctuations for segmentation);

Step2(Sub-sentence Parsing): Parsing each sub-sentence deterministically;

Step3(Parsing Combination): Finding main-root among all the sub-roots, then combining the dependency structure of sub-sentences by making main-root as the head of both the left sub-roots and the punctuations for sentence segmentation.

2 Sentence Segmentation

As mentioned in section 1, the punctuations depending on main-root can be used to segment a sentence into several sub-sentences, whose sub-root depends on main-root or is main-root. But by analysis, we found only several punctuations were used as separator commonly. To ensure the accuracy of sentence segmentation, we first define the punctuations which are possible for segmentation as **valid punctuation**, which includes *comma, period, colon, semicolon, question mark, exclamatory mark and ellipsis*. Then the task in step 1 is to find punctuations which are able to segment a sentence from all the valid punctuations in a sentence, and use them to divide the sentence into two or more sub-sentences.

We define a classifier (called as **sentence segmenter**) to classify the valid punctuations in a sentence to be good or bad for sentence segmentation. SVM (Sebastiani, 2002) is selected as classification model for its robustness to over-fitting and high performance.

Table 1 shows the binary features defined for sentence segmentation. We use a lexicon consisting of all the words in Penn Chinese Treebank 5.1 to lexicalize word features. For example, if word 为 (for) is the 27150_{th} word in the lexicon, then feature $Word_1$ of PU_B (see Figure 2) is '27150:1'. The pos-tag features are got in the same way by a pos-tag list containing 33 pos-tags, which follow the definition in Penn Chinese Treebank. Such method is also used to get word and pos-tag features in other modules.

Table 1. Features for sentence segmenter

Feature	Description
$Word_n/Pos_n$	word/pos-tag in different position, n=-2,-1,0,1,2
Word_left/ Pos_left	word/pos-tag between the first left valid punctuation and current punctuation
Word_right/ Pos_right	word/pos-tag between current punctuation and the first right valid punctuation
#Word_left/ #Word_right	if the number of words between the first left/right valid punctuation and current punctuation is higher than 2, set as 1; otherwise set as 0
V_left/ V_right	if there is a verb between the first left/right valid punctuation and current punctuation, set as 1; otherwise set as 0
N_leftFirst/ N_rightFirst	if the left/right neighbor word is a noun, set as 1; otherwise set as 0
P_rightFirst/ CS_rightFirst	if the right neighbor word is a preposition/subordinating conjunction, set as 1; otherwise set as 0

3 Sub-sentence Parsing

3.1 Parsing Algorithm

The parsing algorithm in step 2 is a shift-reduce parser based on (Yamada and Matsumoto, 2003). We call it as **sub-sentence parser**.

Two stacks *P* and *U* are defined, where stack *P* keeps the words under consideration and stack *U* remains all the unparsed words. All the dependency relations created by the parser are stored in queue *A*.

At start, stack *P* and queue *A* are empty and stack *U* contains all the words. Then word on the top of stack *U* is pushed into stack *P*, and a trained classifier finds probable action for word pair <*p*,*u*> on the top of the two stacks. After that, according to different actions, dependency relations are created and pushed into queue *A*, and the elements in the two stacks move at the same time. Parser stops when stack *U* is empty and the dependency tree can be drawn according to the relations stored in queue *A*.

Four actions are defined for word pair <*p*, *u*>:

LEFT: if word *p* modifies word *u*, then push $p \rightarrow u$ into *A* and push *u* into *P*.

RIGHT: if word *u* modifies word *p*, then push $u \rightarrow p$ into *A* and pop *p*.

REDUCE: if there is no word *u'* ($u' \in U$ and $u' \neq u$) which modifies *p*, and word next to *p* in stack *P* is *p*'s head, then pop *p*.

SHIFT: if there is no dependency relation between *p* and *u*, and word next to *p* in stack *P* is not *p*'s head, then push *u* into stack *P*.

We construct a classifier for each action separately, and classify each word pair by all the classifiers. Then the action with the highest classification score is selected. SVM is used as the classifier, and *One vs. All* strategy (Berger, 1999) is applied for its good efficiency to extend binary classifier to multi-class classifier.

3.2 Features

Features are crucial to this step. First, we define some features based on local context (see F_{local} in Table 2), which are often used in other deterministic parsers (Yamada and Matsumoto, 2003; Nivre et al., 2006). Then, to get top-down information, we add some global features (see F_{global} in Table 2). All the features are binary features, except that *Distance* is normalized between 0-1 by the length of sub-sentence.

Before parsing, we use a root finder (i.e. the sub-sentence root finder introduced in Section 4) to get $Root_n$ feature, and develop a **baseNP chunker** to get $BaseNP_n$ feature. In the baseNP chunker, *IOB* representation is applied for each word, where *B* means the word is the beginning of a baseNP, *I* means the word is inside of a baseNP, and *O* means the word is outside of a baseNP. Tagging is performed by SVM with *One vs. All* strategy. Features used in baseNP chunking are current word, surrounding words and their corresponding pos-tags. Window size is 5.

Table 2. Features for sub-sentence parser

Feature		Description
Local Feature (F_{local})	$Word_n$/ Pos_n	word/pos-tag in different position, n= P_0, P_1, P_2, U_0, U_1, U_2 (P_i/U_i mean the i_{th} position from top in stack *P*/*U*)
	$Word\_child_n$/ Pos_child_n	the word/pos-tag of the children of $Word_n$, n= P_0, P_1, P_2, U_0, U_1, U_2
	Distance	distance between p and u in sentence
Global Feature (F_{global})	$Root_n$	if $Word_n$ is the sub-root of this sub-sentence, set as 1; otherwise set as 0
	$BaseNP_n$	baseNP tag of $Word_n$

Table 3. Features for sentence/sub-sentence root finder

Feature	Description
$Word_n$/Pos_n	words in different position, n=-2,-1,0,1,2
Word_left/Pos_left	$word_n$/pos_n where n<-2
Word_right/Pos_right	$word_n$/pos_n where n>2
#Word_left/ #Word_right	if the number of words between the start/end of sentence and current word is higher than 2, set as 1; otherwise set as 0
V_left/V_right	if there is a verb between the start/end of sentence and current word, set as 1; otherwise set as 0
$Noun_n$/$Verb_n$/Adj_n	if the word in different position is a noun/verb/adjective, set as 1; otherwise set as 0. n=-2,-1,1,2
Dec_right	if the word next to current word in right side is 的(of), set as 1; otherwise set as 0
CC_left	if there is a coordinating conjunction between the start of sentence and current word, set as 1; otherwise set as 0
$BaseNP_n$	baseNP tag of $Word_n$

4 Parsing Combination

A root finder is developed to find main-root for parsing combination. We call it as **sentence root finder**. We also retrain the same module to find the sub-root in step 2, and call it as **sub-sentence root finder**.

We define the root finding problem as a classification problem. A classifier, where we still select SVM, is trained to classify each word to be root or not. Then the word with the highest classification score is chosen as root. All the binary features for root finding are listed in Table 3. Here the baseNP chunker introduced in section 3.2 is used to get the $BaseNP_n$ feature.

5 Experimental Results

5.1 Data Set and Experimental Setting

We use Penn Chinese Treebank 5.1 as data set. To transfer the phrase structure into dependency structure, head rules are defined based on Xia's head percolation table (Xia and Palmer, 2001). 16,984 sentences and 1,292 sentences are used for training and testing. The same training data is also used to train the sentence segmenter, the baseNP chunker, the sub-sentence root finder, and the sentence root finder. During both training and testing, the gold-standard word segmentation and pos-tag are applied.

TinySVM is selected as a SVM toolkit. We use a polynomial kernel and set the degree as 2 in all the experiments.

5.2 Three-step Parsing vs. One-step Parsing

First, we evaluated the dependency accuracy and root accuracy of both three-step parsing and one-step parsing. Three-step parsing is the proposed parser and one-step parsing means parsing a sentence in sequence (i.e. only using step 2). Local and global features are used in both of them.

Results (see Table 4) showed that because of the shortening of sentence length and the prevention of error propagation three-step parsing got 2.14% increase on dependency accuracy compared with one-step parsing. Based on McNemar's test (Gillick and Cox, 1989), this improvement was considered extremely statistically significant ($p<0.0001$). In addition, the proposed parser got 1.01% increase on root accuracy.

Table 4. Parsing result of three-step and one-step parsing

Parsing Strategy	Dep.Accu. (%)	Root Accu. (%)	Avg. Parsing Time (sec.)
One-step Parsing	82.12	74.92	22.13
Three-step Parsing	84.26 (+2.14)	75.93 (+1.01)	24.27 (+2.14)

Then we tested the average parsing time for each sentence to verify the efficiency of proposed parser. The average sentence length is 21.68 words. Results (see Table 4) showed that compared with one-step parsing, the proposed parser only used 2.14 more seconds aver-

agely when parsing one sentence, which did not affect efficiency greatly.

To verify the effectiveness of proposed parser on complex sentences, which contain two or more sub-sentences according to our definition, we selected 665 such sentences from testing data set and did evaluation again. Results (see Table 5) proved that our parser outperformed one-step parsing successfully.

Table 5. Parsing result of complex sentence

Parsing Strategy	Dep.Accu. (%)	Root Accu. (%)
One-step Parsing	82.56	78.95
Three-step Parsing	84.94 (+2.38)	79.25 (+0.30)

5.3 Comparison with Others' Work

At last, we compare the proposed parser with Nivre's parser (Hall et al., 2006). We use the same head rules for dependency transformation as what were used in Nivre's work. We also used the same training (section 1-9) and testing (section 0) data and retrained all the modules. Results showed that the proposed parser achieved 84.50% dependency accuracy, which was 0.20% higher than Nivre's parser (84.30%).

6 Discussion

In the proposed parser, we used five modules: sentence segmenter (step1); sub-sentence root finder (step2); baseNP chunker (step2&3); sub-sentence parser (step2); and sentence root finder (step3).

The robustness of the modules will affect parsing accuracy. Thus we evaluated each module separately. Results (see Table 6) showed that all the modules got reasonable accuracy except for the sentence root finder. Considering about this, in step 3 we found main-root only from the sub-roots created by step 2. Because the sub-sentence parser used in step 2 had good accuracy, it could provide relatively correct candidates for main-root finding. Therefore it helped decrease the influence of the poor sentence root finding to the proposed parser.

Table 6. Evaluation result of each module

Module	F-score(%)	Dep.Accu(%)
Sentence Segmenter (M1)	88.04	---
Sub-sentence Root Finder (M2)	88.73	---
BaseNP Chunker (M3)	89.25	---
Sub-sentence Parser (M4)	---	85.56
Sentence Root Finder (M5)	78.01	---

Then we evaluated the proposed parser assuming using gold-standard modules (except for sub-sentence parser) to check the contribution of each module to parsing. Results (see Table 7) showed that (1) the accuracy of current sentence segmenter was acceptable because only small increase on dependency accuracy and root accuracy was got by using gold-standard sentence segmentation; (2) the correct recognition of baseNP could help improve dependency accuracy but gave a little contribution to root accuracy; (3) the accuracy of both sub-sentence root finder and sentence root finder was most crucial to parsing. Therefore improving the two root finders is an important task in our future work.

Table 7. Parsing result with gold-standard modules

Gold-standard Module	Dep.Accu(%)	Root.Accu(%)
w/o	84.26	75.93
M1	84.51	76.24
M1+M2	86.57	80.34
M1+M2+M3	88.63	80.57
M1+M2+M3+M5	91.25	91.02

7 Conclusion and Future Work

We propose a three-step deterministic dependency parser for parsing Chinese. It aims to solve the error propagation problem by dividing a sentence into independent parts and parsing them separately. Results based on Penn Chinese Treebank 5.1 showed that compared with the deterministic parser which parsed a sentence in sequence, the proposed parser achieved extremely significant increase on dependency accuracy.

Currently, the proposed parser is designed only for Chinese. But we believe it can be easily adapted to other languages because no language-limited information is used. We will try this work in the future. In addition, improving sub-sentence root finder and sentence root finder will also be considered in the future.

Acknowledgement

We would like to thank Dr. Daisuke Kawahara and Dr. Eiji Aramaki for their helpful discussions. We also thank the three anonymous reviewers for their valuable comments.

Reference

A.Berger. Error-correcting output coding for text classification. 1999. In *Proceedings of the IJCAI-99 Workshop on Machine Learning for Information Filtering*.

M.Chang, Q.Do and D.Roth. 2006. A Pipeline Framework for Dependency Parsing. In *Proceedings of Coling-ACL 2006*.

Y.Cheng, M.Asahara and Y.Matsumoto. 2005. Chinese Deterministic Dependency Analyzer: Examining Effects of Global Features and Root Node Finder. In *Proceedings of IJCNLP 2005*.

L.Gillick and S.J.Cox. 1989. Some Statistical Issues in the Comparison of Speech Recognition Algorithms. In *Proceedings of ICASSP*.

J.Hall, J.Nivre and J.Nilsson. 2006. Discriminative Classifiers for Deterministic Dependency Parsing. In *Proceedings of Coling-ACL 2006*. pp. 316-323.

J.Nivre and M.Scholz. 2004. Deterministic Dependency Parsing of English Text. In *Proceedings of Coling 2004*. pp. 64-70.

F.Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1): 1-47.

F.Xia and M.Palmer. 2001. Converting Dependency Structures to Phrase Structures. In *HLT-2001*.

N.Xue, F.Chiou and M.Palmer. 2002. Building a Large-Scale Annotated Chinese Corpus. In P*roceedings of COLING 2002*.

H.Yamada and Y.Matsumoto. 2003. Statistical Dependency Analysis with Support Vector Machines. In *Proceedings of IWPT. 2003*.

Comparing Wikipedia and German Wordnet by Evaluating Semantic Relatedness on Multiple Datasets

Torsten Zesch and **Iryna Gurevych** and **Max Mühlhäuser**
Ubiquitous Knowledge Processing Group, Telecooperation Division
Darmstadt University of Technology, D-64289 Darmstadt, Germany
{zesch,gurevych,max} (at) tk.informatik.tu-darmstadt.de

Abstract

We evaluate semantic relatedness measures on different German datasets showing that their performance depends on: (i) the definition of relatedness that was underlying the construction of the evaluation dataset, and (ii) the knowledge source used for computing semantic relatedness. We analyze how the underlying knowledge source influences the performance of a measure. Finally, we investigate the combination of wordnets and Wikipedia to improve the performance of semantic relatedness measures.

1 Introduction

Semantic similarity (**SS**) is typically defined via the lexical relations of synonymy ($automobile - car$) and hypernymy ($vehicle - car$), while **semantic relatedness** (**SR**) is defined to cover any kind of lexical or functional association that may exist between two words. Many NLP applications, like sense tagging or spelling correction, require knowledge about semantic relatedness rather than just similarity (Budanitsky and Hirst, 2006). For these tasks, it is not necessary to know the exact type of semantic relation between two words, but rather if they are closely semantically related or not. This is also true for the work presented herein, which is part of a project on electronic career guidance. In this domain, it is important to conclude that the words "baker" and "bagel" are closely related, while the exact type of a semantic relation does not need to be determined.

As we work on German documents, we evaluate a number of SR measures on different German datasets. We show that the performance of measures strongly depends on the underlying knowledge source. While WordNet (Fellbaum, 1998) models SR, wordnets for other languages, such as the German wordnet GermaNet (Kunze, 2004), contain only few links expressing SR. Thus, they are not well suited for estimating SR.

Therefore, we apply the Wikipedia category graph as a knowledge source for SR measures. We show that Wikipedia based SR measures yield better correlation with human judgments on SR datasets than GermaNet measures. However, using Wikipedia also leads to a performance drop on SS datasets, as knowledge about classical taxonomic relations is not explicitly modeled. Therefore, we combine GermaNet with Wikipedia, and yield substantial improvements over measures operating on a single knowledge source.

2 Datasets

Several German datasets for evaluation of SS or SR have been created so far (see Table 1). Gurevych (2005) conducted experiments with a German translation of an English dataset (Rubenstein and Goodenough, 1965), but argued that the dataset (**Gur65**) is too small (it contains only 65 noun pairs), and does not model SR. Thus, she created a German dataset containing 350 word pairs (**Gur350**) containing nouns, verbs and adjectives that are connected by classical and non-classical relations (Morris and Hirst, 2004). However, the dataset is biased towards strong classical relations, as word pairs were manually selected. Thus, Zesch and Gurevych (2006) semi-automatically created word pairs from domain-specific corpora. The resulting **ZG222** dataset contains 222 word pairs that are connected by all kinds of lexical semantic relations. Hence, it is particularly suited for analyzing the capability of a measure to estimate SR.

Proceedings of NAACL HLT 2007, Companion Volume, pages 205–208,
Rochester, NY, April 2007.

Dataset	Year	Language	# Pairs	POS	Type	Scores	# Subjects	Correlation r Inter	Intra
Gur65	2005	German	65	N	SS	discrete {0,1,2,3,4}	24	.810	-
Gur350	2006	German	350	N, V, A	SR	discrete {0,1,2,3,4}	8	.690	-
ZG222	2006	German	222	N, V, A	SR	discrete {0,1,2,3,4}	21	.490	.647

Table 1: Comparison of datasets used for evaluating semantic relatedness.

3 Semantic Relatedness Measures

Semantic wordnet based measures Lesk (1986) introduced a measure (***Les***) based on the number of word overlaps in the textual definitions (or glosses) of two terms, where higher overlap means higher similarity. As GermaNet does not contain glosses, this measure cannot be employed. Gurevych (2005) proposed an alternative algorithm (***PG***) generating surrogate glosses by using a concept's relations within the hierarchy. Following the description in Budanitsky and Hirst (2006), we further define several measures using the taxonomy structure.

$$sim_{PL} = l(c_1, c_2)$$
$$sim_{LC} = -\log \frac{l(c_1, c_2)}{2 \times depth}$$
$$sim_{Res} = IC(c_i) = -\log(p(lcs(c_1, c_2)))$$
$$dist_{JC} = IC(c_1) + IC(c_2) - 2IC(lcs(c_1, c_2))$$
$$sim_{Lin} = 2 \times \frac{IC(lcs(c_1, c_2))}{IC(c_1) + IC(c_2)}$$

PL is the taxonomic path length $l(c_1, c_2)$ between two concepts $c1$ and $c2$. ***LC*** normalizes the path length with the depth of the taxonomy. ***Res*** computes SS as the information content (IC) of the lowest common subsumer (lcs) of two concepts, while ***JC*** combines path based and IC features.[1] ***Lin*** is derived from information theory.

Wikipedia based measures For computing the SR of two words w_1 and w_2 using Wikipedia, we first retrieve the articles or disambiguation pages with titles that equal w_1 and w_2 (see Figure 1). If we hit a redirect page, we retrieve the corresponding article or disambiguation page instead. In case of an article, we insert it into the candidate article set (A_1 for $w1$, A_2 for w_2). In case of a disambiguation page, the page contains links to all encoded word senses, but it may also contain other links. Therefore, we only consider links conforming to the pattern ⟨Title_(DisambiguationText)⟩[2] (e.g. "Train_(roller coaster)"). Following all such links gives the candidate article set. If no disambiguation links are found, we take the first link on the page, as most important links tend to come first. We add the corresponding articles to the candidate set. We form pairs from each candidate article $a_i \in A_1$ and each article $a_j \in A_2$. We then compute $SR(a_i, a_j)$ for each pair. The output of the algorithm is the maximum SR value $\max_{a_i \in A_1, a_j \in A_2}(SR(a_i, a_j))$.[3]

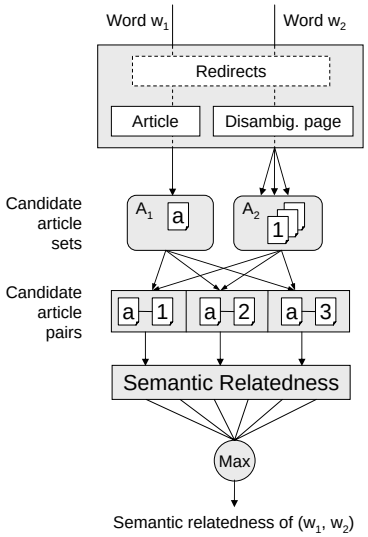


Figure 1: Steps for computing SR using Wikipedia.

As most SR measures have been developed for taxonomic wordnets, porting them to Wikipedia requires some modifications (see Figure 2). Text overlap measures can be computed based on the article text, while path based measures operate on the category graph. We compute the overlap between article

[1] Note that JC returns a distance value instead of a similarity value resulting in negative correlation with human judgments.

[2] '_(DisambiguationText)' is optional.

[3] Different from our approach, Strube and Ponzetto (2006) use a disambiguation strategy that returns only a single candidate article pair. This unnecessarily limits a measure's potential to consider SR between all candidate article pairs. They also limit the search for a lcs to a manually specified threshold of 4.

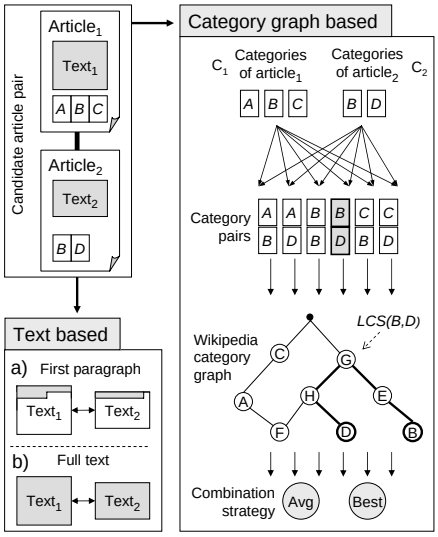


Figure 2: SR measures adapted on Wikipedia.

texts based on (i) the first paragraph, as it usually contains a short gloss, and (ii) the full article text. As Wikipedia articles do not form a taxonomy, path based measures have to be adapted to the Wikipedia category graph (see the right part of Figure 2). We define C_1 and C_2 as the set of categories assigned to article a_i and a_j, respectively. We compute the SR value for each category pair (c_k, c_l) with $c_k \in C_1$ and $c_l \in C_2$. We use two different strategies to combine the resulting SR values: First, we choose the best value among all pairs (c_k, c_l), i.e., the minimum for path based, and the maximum for information content based measures. As a second strategy, we average over all category pairs.

4 Experiments & Results

Table 2 gives an overview of our experimental results on three German datasets. Best values for each dataset and knowledge source are in bold. We use the PG measure in optimal configuration as reported by Gurevych (2005). For the Les measure, we give the results for considering: (i) only the first paragraph (+First) and (ii) the full text (+Full). For the path length based measures, we give the values for averaging over all category pairs (+Avg), or taking the best SR value computed among the pairs (+Best). For each dataset, we report Pearson's correlation r with human judgments on pairs that are found in **both resources** (**BOTH**). Otherwise, the results would not be comparable. We additionally use a subset containing only **noun-noun pairs** (**BOTH NN**). This comparison is fairer, because article titles in Wikipedia are usually nouns. Table 2 also gives the inter annotator agreement for each subset. It constitutes an upper bound of a measure's performance.

Our results on Gur65 using GermaNet are very close to those published by Gurevych (2005), ranging from 0.69–0.75. For Gur350, the performance drops to 0.38–0.50, due to the lower upper bound, and because GermaNet does not model SR well. These findings are endorsed by an even more significant performance drop on ZG222. The measures based on Wikipedia behave less uniformly. Les yields acceptable results on Gur350, but is generally not among the best performing measures. LC +Avg yields the best performance on Gur65, but is outperformed on the other datasets by PL +Best, which performs equally good for all datasets.

If we compare GermaNet based and Wikipedia based measures, we find that the knowledge source has a major influence on performance. When evaluated on Gur65, that contains pairs connected by SS, GermaNet based measures perform near the upper bound and outperform Wikipedia based measures by a wide margin. On Gur350 containing a mix of SS and SR pairs, most measures perform comparably. Finally, on ZG222, that contains pairs connected by SR, the best Wikipedia based measure outperforms all GermaNet based measures.

The impressive performance of PL on the SR datasets cannot be explained with the structural properties of the category graph (Zesch and Gurevych, 2007). Semantically related terms, that would not be closely related in a taxonomic wordnet structure, are very likely to be categorized under the same Wikipedia category, resulting in short path lengths leading to high SR. These findings are contrary to that of (Strube and Ponzetto, 2006), where LC outperformed path length. They limited the search depth using a manually defined threshold, and did not compute SR between all candidate article pairs.

Our results show that judgments on the performance of a measure must always be made with respect to the task at hand: computing SS or SR. Depending on this decision, we can choose the best underlying knowledge source. GermaNet is better for

		GUR65	GUR350		ZG222	
		BOTH NN	BOTH	BOTH NN	BOTH	BOTH NN
# Word Pairs		53	116	91	55	45
Inter Annotator Agreement		0.80	0.64	0.63	0.44	0.43
GermaNet	PG	0.69	0.38	0.42	**0.23**	0.21
	JC	**-0.75**	**-0.52**	-0.48	-0.19	**-0.25**
	Lin	0.73	0.50	**0.50**	0.08	-0.12
	Res	0.71	0.42	0.42	0.10	0.13
Wikipedia	Les +First	0.16	0.36	0.32	0.01	0.11
	Les +Full	0.19	0.34	0.37	0.13	0.17
	PL +Avg	-0.32	-0.34	-0.46	-0.36	-0.43
	PL +Best	-0.35	**-0.42**	**-0.53**	**-0.43**	**-0.49**
	LC +Avg	**0.37**	0.25	0.34	0.30	0.30
	LC +Best	0.21	0.12	0.21	0.15	0.12
Combination	Linear	**0.77**	**0.59**	**0.60**	0.38	0.43
	POS	-	0.55	-	**0.48**	-

Table 2: Correlation r of human judgments with SR measures on different datasets.

estimating SS, while Wikipedia should be used to estimate SR. Therefore, a measure based on a single knowledge source is unlikely to perform well in all settings. We computed a linear combination of the best measure from GermaNet and from Wikipedia. Results for this experiment are labeled *Linear* in Table 2. *POS* is an alternative combination strategy, where Wikipedia is only used for noun-noun pairs. GermaNet is used for all other part-of-speech (POS) combinations. For most datasets, we find a combination strategy that outperforms all single measures.

5 Conclusion

We have shown that in deciding for a specific measure and knowledge source it is important to consider (i) whether the task at hand requires SS or SR, and (ii) which POS are involved. We pointed out that the underlying knowledge source has a major influence on these points. GermaNet is better used for SS, and contains nouns, verbs, and adjectives, while Wikipedia is better used for SR between nouns. Thus, GermaNet and Wikipedia can be regarded as complementary. We have shown that combining them significantly improves the performance of SR measures up to the level of human performance.

Future research should focus on improving the strategies for combining complementary knowledge sources. We also need to evaluate a wider range of measures to validate our findings. As the simple PL measure performs remarkably well, we should also consider computing SR based on the Wikipedia article graph instead of the category graph.

Acknowledgments

This work was supported by the German Research Foundation under grant "Semantic Information Retrieval from Texts in the Example Domain Electronic Career Guidance", GU 798/1-2.

References

Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based Measures of Semantic Distance. *Computational Linguistics*, 32(1).

Christiane Fellbaum. 1998. *WordNet An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Iryna Gurevych. 2005. Using the Structure of a Conceptual Network in Computing Semantic Relatedness. In *Proc. of IJCNLP*, pages 767–778.

Claudia Kunze, 2004. *Lexikalisch-semantische Wortnetze*, chapter Computerlinguistik und Sprachtechnologie, pages 423–431. Spektrum Akademischer Verlag.

Michael Lesk. 1986. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to tell a pine cone from an ice cream cone. In *Proc. of the 5th Annual International Conference on Systems Documentation*, pages 24–26.

Jane Morris and Graeme Hirst. 2004. Non-Classical Lexical Semantic Relations. In *Proc. of the Workshop on Computational Lexical Semantics, NAACL-HTL.*

Herbert Rubenstein and John B. Goodenough. 1965. Contextual Correlates of Synonymy. *Communications of the ACM*, 8(10):627–633.

Michael Strube and Simone Paolo Ponzetto. 2006. WikiRelate! Computing Semantic Relatedness Using Wikipedia. In *Proc. of AAAI*, pages 1219–1224.

Torsten Zesch and Iryna Gurevych. 2006. Automatically Creating Datasets for Measures of Semantic Relatedness. In *Proc. of the Workshop on Linguistic Distances, ACL*, pages 16–24.

T. Zesch and I. Gurevych. 2007. Analysis of the Wikipedia Category Graph for NLP Applications. In *Proc. of the TextGraphs-2 Workshop, NAACL-HLT*, (to appear).

Selective Phrase Pair Extraction for Improved Statistical Machine Translation

Luke S. Zettlemoyer
MIT CSAIL
Cambridge, MA 02139
lsz@csail.mit.edu

Robert C. Moore
Microsoft Research
One Microsoft Way
Redmond, WA 98052
bobmoore@microsoft.com

Abstract

Phrase-based statistical machine translation systems depend heavily on the knowledge represented in their phrase translation tables. However, the phrase pairs included in these tables are typically selected using simple heuristics that potentially leave much room for improvement. In this paper, we present a technique for selecting the phrase pairs to include in phrase translation tables based on their estimated quality according to a translation model. This method not only reduces the size of the phrase translation table, but also improves translation quality as measured by the BLEU metric.

1 Introduction

Phrase translation tables are the heart of phrase-based statistical machine translation (SMT) systems. They provide pairs of phrases that are used to construct a large set of potential translations for each input sentence, along with feature values associated with each phrase pair that are used to select the best translation from this set.[1]

The most widely used method for building phrase translation tables (Koehn et al., 2003) selects, from a word alignment of a parallel bilingual training corpus, all pairs of phrases (up to a given length) that are consistent with the alignment. This procedure typically generates many phrase pairs that are not remotely reasonable translation candidates.[2] To avoid creating translations that use these pairs, a set of features is computed for each pair. These features are used to train a translation model, and phrase pairs that produce low scoring translations are avoided. In practice, it is often assumed that current translation models are good enough to avoid building translations with these unreasonable phrase pairs.

In this paper, we question this assumption by investigating methods for pruning low quality phrase pairs. We present a simple procedure that reduces the overall phrase translation table size while increasing translation quality. The basic idea is to initially gather the phrase pairs and train an translation model as usual, but to then select a subset of the overall phrases that performs the best, prune the others, and retrain the translation model. In experiments, this approach reduced the size of the phrase tranlsation table by half, and improved the BLEU score of the resulting translations by up to 1.5 points.

2 Background

As a baseline, we present a relatively standard SMT approach, following Koehn et al. (2003). Potential translations are scored using a linear model where the best translation is computed as

$$\arg\max_{t,a} \sum_{i=1}^{n} \lambda_i f_i(s, a, t)$$

where s is the input sentence, t is the output sentence, and a is a phrasal alignment that specifies how

[1] A "phrase" in this sense can be any contiguous sequence of words, and need not be a complete linguistic constituent.

[2] In one experiment, we managed to generate more than 117,000 English phrases for the the French word "de".

Proceedings of NAACL HLT 2007, Companion Volume, pages 209–212,
Rochester, NY, April 2007.

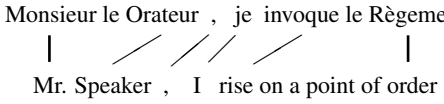


Figure 1: A word aligned sentence pair.

t is constructed from s. The weights λ_i associated with each feature f_i are tuned to maximize the quality of the translations.

The training procedure starts by computing a word alignment for each sentence pair in the training corpus. A word alignment is a relation between the words in two sentences where, intuitively, words are aligned to their translation in the other language. In this work, we use a discriminatively trained word aligner (Moore et al., 2006) that has state of the art performance. Figure 1 presents a high quality alignment produced by this aligner.

Given a word aligned corpus, the second step is to extract a phrase translation table. Each entry in this table contains a source language phrase s, a target language phrase t, and a list of feature values $\phi(s,t)$. It is usual to extract every phrase pair, up to a certain phrase length, that is consistent with the word alignment that is annotated in the corpus. Each consistent pair must have at least one word alignment between words within the phrases and no words in either phrase can be aligned any words outside of the phrases. For example, Figure 2 shows some of the phrase pairs that would be extracted from the word-aligned sentence pair in Figure 1. A full list using phrases of up to three words would include 28 pairs.

For each extracted phrase pair (s,t), feature values $\phi(s,t) = \langle \log p(s|t), \log p(t|s), \log l(s,t) \rangle$ are computed. The first two features, the log translation and inverse translation probabilities, are estimated by counting phrase cooccurrences, following Koehn et al. (2003). The third feature is the logarithm of a lexical score $l(s,t)$ that provides a simple form of smoothing by weighting a phrase pair based on how likely individual words within the phrases are to be translations of each other. We use a version from Foster et al. (2006), modified from (Koehn et al., 2003), which is an average of pairwise word translation probabilities.

In phrase-based SMT, the decoder produces translations by dividing the source sentence into a sequence of phrases, choosing a target language phrase

#	Source Lang. Phrase	Target Lang. Phrase
1	Monsieur	Mr.
2	Monsieur le	Mr.
3	Monsieur le Orateur	Mr. Speaker
4	le Orateur	Speaker
5	Orateur	Speaker
...	...	...
23	le Règlement	point of order
24	le Règlement	of order
25	le Règlement	order
26	Règlement	point of order
27	Règlement	of order
28	Règlement	order

Figure 2: Phrase pairs consistent with the word alignment in Figure 1.

as a translation for each source language phrase, and ordering the target language phrases to build the final translated sentence. Each potential translation is scored according to a weighted linear model. We use the three features from the phrase translation table, summing their values for each phrase pair used in the translation. We also use four additional features: a target language model, a distortion penalty, the target sentence word count, and the phrase pair count, all computed as described in (Koehn, 2004). For all of the experiments in this paper, we used the Pharaoh beam-search decoder (Koehn, 2004) with the features described above.

Finally, to estimate the parameters λ_i of the weighted linear model, we adopt the popular minimum error rate training procedure (Och, 2003) which directly optimizes translation quality as measured by the BLEU metric.

3 Selective Phrase Pair Extraction

In order to improve performance, it is important to select high quality phrase pairs for the phrase translation table. We use two key ideas to guide selection:

- **Preferential Scoring:** Phrase pairs are selected using a function $q(s,t)$ that returns a high score for source, target phrase pairs (s,t) that lead to high quality translations.
- **Redundancy Constraints:** Our intuition is that each occurrence of a source or target language phrase really has at most one translation for that sentence pair. Redundancy constraints minimize the number of possible translations that are extracted for each phrase occurrence.

Selecting phrases that a translation model prefers and eliminating at least some of the ambiguity that comes with extracting multiple translations for a single phrase occurrence creates a smaller phrase translation table with higher quality entries.

The ideal scoring metric would give high scores to phrase pairs that lead to high-quality translations and low scores to those that would decrease translation quality. The best such metric we have available is provided by the overall translation model. Our scoring metric $q(s,t)$ is therefore computed by first extracting a full phrase translation table, then training a full translation model, and finally using a subpart of the model to score individual phrase pairs in isolation. Because the scoring is tied to a model that is optimized to maximize translation quality, more desirable phrase pairs should be given higher scores.

More specifically, $q(s,t) = \phi(s,t) \cdot \lambda$ where $\phi(s,t)$ is the length three vector that contains the feature values stored with the phrase pair (s,t) in the phrase translation table, and λ is a vector of the three parameter values that were learned for these features by the full translation model. The rest of the features are ignored because they are either constant or depend on the target language sentence which is fixed during phrase extraction. In essence, we are using the subpart of a full translation model that looks at phrase pair identity and scoring the pair based on how the full model would like it.

This scoring metric is used in a phrase pair selection algorithm inspired by competitive linking for word alignment (Melamed, 2000). *Local competitive linking* extracts high scoring phrase pairs while enforcing a redundancy constraint that minimizes the number of phrase pairs that share a common phrase. For each sentence pair in the training set, this algorithm marks the highest scoring phrase pair, according to $q(s,t)$, containing each source language phrase and the highest scoring phrase pair containing each target language phrase. Each of these marked phrase pairs is selected and the phrase translation table is rebuilt. This is a soft redundancy constraint because a phrase pair will only be excluded if there is a higher scoring pair that shares its source language phrase and a higher scoring pair that shares its target language phrase. For example, consider again the phrase pairs in Figure 2 and assume they are sorted by their scores. Local competitive linking will select every phrase pair except for 27 and 28. All other pairs are the highest scoring options for at least one of their phrases.

Selective phrase extraction with competitive linking can be seen as a Viterbi reestimation algorithm. Because we are extracting fewer phrase pairs, the features associated with each phrase pair will differ. If the removed phrases were not real translations of each other in the first place, the translation features $p(s|t)$ and $p(t|s)$ should be better estimates because the high quality phrases that remain will be given the probability mass that was assigned to the pruned phrase pairs. Although we are running it in a purely discriminative setting, it has a similar feel to an EM algorithm. First, a full phrase translation table and parameter estimate is computed. Then, based on that estimate, a subset of the phrases is selected which, in turn, supplies a new estimate for the parameters. One question is how many times to run this reestimation procedure. We found, on the development set, that it never helped to run more than one iteration. Perhaps because of the hard nature of the algorithm, repeated iterations caused slight decreases in phrase translation table size and overall performance.

4 Experiments

In this section, we report experiments conducted with Canadian Hansards data from the 2003 HLT-NAACL word-alignment workshop (Mihalcea and Pedersen, 2003). Phrase pairs are extracted from 500,000 word-aligned French-English sentence pairs. Translation quality is evaluated according to the BLEU metric (with one reference translation). Three additional disjoint data sets (from the same source) were used, one with 500 sentence pairs for minimum error rate training, another with 1000 pairs for development testing, and a final set of 2000 sentence pairs for the final test. For each experiment, we trained the full translation model as described in Section 2. Each trial varied only in the phrase translation table that was used.[3]

One important question is what the maximum phrase length should be for extraction. To investigate this issue, we ran experiments on the devel-

[3]These experiments also used the default pruning from the Pharaoh decoder, allowing only the 10 best output phrases to be considered for each input phrase. This simple global pruning cannot be substituted for the competitive linking described here.

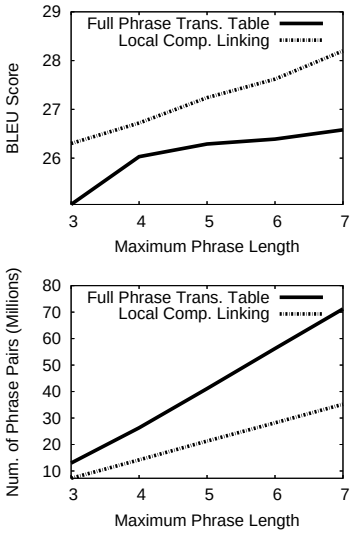


Figure 3: Scaling the maximum phrase length.

opment set. Figure 3 shows a comparison of the full phrase table to local competitive linking as the maximum phrase length is varied. Local competitive linking consistently outperforms the full table and the difference in BLEU score seems to increase with the length. The growth in the size of the phrase translation table seems to be linear with maximum phrase length in both cases, with the table size growing at a slower rate under local competitive linking.

To verify these results, we tested the model trained with the full phrase translation table against the model trained with the table selected by local competitive linking on the heldout test data. Both tables included phrases up to length 7 and the models were tested on a set of 2000 unseen sentence pairs. The results matched the development experiments. The full system scored 26.78 while the local linking achieved 28.30, a difference of 1.52 BLEU points.

5 Discussion

The most closely related work attempts to create higher quality phrase translation tables by learning a generative model that directly incorporates phrase pair selection. The original approach (Marcu and Wong, 2002) was limited due to computational constraints but recent work (DeNero et al., 2006; Birch et al., 2006) has improved the efficiency by using word alignments as constraints on the set of possible phrase pairs. The best results from this line of work allow for a significantly smaller phrase translation table, but never improve translation performance.

In this paper, we presented an algorithm that improves translation quality by selecting a smaller phrase translation table. We hope that this work highlights the need to think carefully about the quality of the phrase translation table, which is the central knowledge source for most modern statistical machine translation systems. The methods used in the experiments are so simple that we believe that there is significant potential for improvement by using better methods for scoring phrase pairs and selecting phrase pairs based those scores.

References

Alexandra Birch, Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Constraining the phrase-based, join probability statistical translation model. In *Proceedings of the Workshop on Stastical Machine Translation*.

John DeNero, Dan Gillick, James Zhang, and Dan Klein. 2006. Why generative phrase models underperform surface heuristics. In *Proceedings of the Workshop on Stastical Machine Translation*.

George Foster, Roland Kuhn, and Howard Johnson. 2006. Phrasetable smoothing for stastical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Stastical phrase-based translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Philipp Koehn. 2004. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *Proceedings of The Sixth Conference of the Association for Machine Translation in the Americas*.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

I. Dan Melamed. 2000. Models of translation equivalence amoung words. *Computational Linguistics*, 26(2):221–249.

Rada Mihalcea and Ted Pedersen. 2003. An evaluation exercise for word alignment. In *Proceedings of the HLT-NAACL 2003 Workshop, Building and Using Parallel Texts: Data Driven Machine Translation and Beyond.*

Robert C. Moore, Wen-tau Yih, and Andreas Bode. 2006. Improved discriminative bilingual word alignment. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics*.

Franz Josef Och. 2003. Minimum error rate training for statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*.

Speech Summarization Without Lexical Features for Mandarin Broadcast News

Jian Zhang
Human Language Technology Center
Electronic and Computer Engineering
University of Science and Technology
Clear Water Bay,Hong Kong
zjustin@ust.hk

Pascale Fung
Human Language Technology Center
Electronic and Computer Engineering
University of Science and Technology
Clear Water Bay,Hong Kong
pascale@ee.ust.hk

Abstract

We present the first known empirical study on speech summarization without lexical features for Mandarin broadcast news. We evaluate acoustic, lexical and structural features as predictors of summary sentences. We find that the summarizer yields good performance at the average F-measure of 0.5646 even by using the combination of acoustic and structural features alone, which are independent of lexical features. In addition, we show that structural features are superior to lexical features and our summarizer performs surprisingly well at the average F-measure of 0.3914 by using only acoustic features. These findings enable us to summarize speech without placing a stringent demand on speech recognition accuracy.

1 Introduction

Speech summarization, a technique of extracting key segments that convey the main content from a spoken document or audio document, has become a new area of study in the last few years. There has been much significant progress made in speech summarization for English or Japanese text and audio sources (Hori and Furui, 2003; Inoue et al., 2004; Koumpis and Renals, 2005; Maskey and Hirschberg, 2003; Maskey and Hirschberg, 2005). Some research efforts have focused on summarizing Mandarin sources (Chen et al., 2006; Huang et al., 2005), which are dependent on lexical features. Considering the difficulty in obtaining high quality transcriptions, some researchers proposed speech summarization systems with non-lexical features (Inoue et al., 2004; Koumpis and Renals, 2005; Maskey and Hirschberg, 2003; Maskey and Hirschberg, 2006). However, there does not exist any empirical study on speech summarization without lexical features for Mandarin Chinese sources. In this paper, we construct our summarizer with acoustic and structural features, which are independent of lexical features, and compare acoustic and structural features against lexical features as predictors of summary sentences.

In Section 2 we review previous work on broadcast news summarization. We describe the Mandarin broadcast news corpus on which our system operates in Section 3. In Section 4 we describe our summarizer and these features used in experiments. We set up our experiments and evaluate the results in Section 5, followed by our conclusion in Section 6.

2 Previous Work

There have been many research efforts on speech summarization. Some methods dependent on lexical features are presented (Inoue et al., 2004; Chen et al., 2006; Huang et al., 2005). (Inoue et al., 2004) uses statistical methods to identify words to include in a summary, based on linguistic and acoustic/prosodic features of the Japanese broadcast news transcriptions; while (Chen et al., 2006) proposes the use of probabilistic latent topical information for extractive summarization of Mandarin spoken documents. (Huang et al., 2005) presents Mandarin spo-

Proceedings of NAACL HLT 2007, Companion Volume, pages 213–216,
Rochester, NY, April 2007.

ken document summarization scheme using acoustic, prosodic, and semantic information. Alternatively, some methods which are independent of lexical features are presented (Maskey and Hirschberg, 2003; Maskey and Hirschberg, 2006). (Maskey and Hirschberg, 2003) extracts structural information from audio documents to help summarization. (Maskey and Hirschberg, 2006) focuses on how to use acoustic information alone to help predict sentences to be included in a summary and shows a novel way of using continuous HMMs for summarizing speech documents without transcriptions.

It is advantageous to build speech summarization models without using lexical features: we can summarize speech data without placing a stringent demand on the speech recognition accuracy. In this paper, we propose one such model on Mandarin broadcast news and compare the effectiveness of acoustic and structural features against lexical features as predictors of summary sentences.

3 The Corpus and Manual Summaries

We use a portion of the 1997 Hub4 Mandarin corpus available via LDC as experiment data. The related audio data were recorded from China Central Television(CCTV) International News programs. They include 23-day broadcast from 14th January, 1997 to 21st April, 1997, which contain 593 stories and weather forecasts. Each broadcast lasts approximately 32 minutes, and has been hand-segmented into speaker turns. For evaluation, we manually annotated these broadcast news, and extracted segments as reference summaries. We divide these broadcast news stories into 3 types: one-turn news, weather forecast, and several-turns news. The content of each several-turn news is presented by more than one reporter, and sometimes interviewees. We evaluate our summarizer on the several-turns news corpus. The corpus has 347 stories which contain 4748 sentences in total.

4 Features and Methodology

4.1 Acoustic/Prosodic Features

Acoustic/prosodic features in speech summarization system are usually extracted from audio data. Researchers commonly use acoustic/prosodic variation – changes in pitch, intensity, speaking rate – and duration of pause for tagging the important contents of their speeches (Hirschberg, 2002). We also use these features for predicting summary sentences on Mandarin broadcast news.

Our acoustic feature set contains thirteen features: ***DurationI***, ***DurationII***, ***SpeakingRate***, ***F0I***, ***F0II***, ***F0III***, ***F0IV***, ***F0V***, ***EI***, ***EII***, ***EIII***, ***EIV*** and ***EV***. *DurationI* is the sentence duration. *DurationII* is the average phoneme duration. General phonetic studies consider that the speaking rate of sentence is reflected in syllable duration. So we use average syllable duration for representing *SpeakingRate*. *F0I* is F0's minimum value. *F0II* is F0's maximum value. *F0III* equals to the difference between *F0II* and *F0I*. *F0IV* is the mean of F0. *F0V* is F0 slope. *EI* is minimum energy value. *EII* is maximum energy value. *EIII* equals to the difference between *EII* and *EI*. *EIV* is the mean of energy value. *EV* is energy slope. We calculate *DurationI* from the annotated manual transcriptions that align the audio documents. We then obtain *DurationII* and *SpeakingRate* by phonetic forced alignment. Next we extract F0 features and energy features from audio data by using Praat (Boersma and Weenink, 1996).

4.2 Structural Features

Each broadcast news of the 1997 Hub4 Mandarin corpus has similar structure, which starts with an anchor, followed by the formal report of the story by other reporters or interviewees.

Our structural feature set consists of 4 features: ***Position***, ***TurnI***, ***TurnII*** and ***TurnIII***. *Position* is defined as follows: one news has *k* sentences, then we set $(1-(0/k))$ as *Position* value of the first sentence in the news, and set $(1-((i-1)/k))$ as *Position* value of the ith sentence. *TurnI* is defined as follows: one news has *m* turns, then we set $(1-(0/m))$ as *TurnI* value of the sentences which belong to the first turn's content, and set $(1-((j-1)/m))$ as *TurnI* values of the sentences which belong to the jth turn's content. *TurnII* is the previous turn's *TurnI* value. *TurnIII* is the next turn's *TurnI* value.

4.3 Reference Lexical Features

Most methods for text summarization mainly utilize lexical features. We are interested in investigating the role of lexical features in comparison to other features. All reference lexical features are extracted

from the manual transcriptions.

Our lexical feature set contains eight features: ***LenI***, ***LenII***, ***LenIII***, ***NEI***, ***NEII***, ***NEIII***, ***TFIDF*** and ***Cosine***. For a sentence, we set the number of words in the sentence as *LenI* value. *LenII* is the previous sentence's *LenI* value. *LenIII* is the next sentence's *LenI* value. For a sentence, we set the number of Named Entities in the sentence as the *NEI* value. We define the number of Named Entities which appear in the sentence at the first time in a news as *NEII* value. *NEIII* value equals to the ratio of the number of unique Named Entities to the number of all Named Entities.

TFIDF is the product of *tf* and *idf*. *tf* is the fraction: the numerator is the number of occurrences of the considered word and the denominator is the number of occurrences of all words in a story. *idf* is the logarithm of the fraction: the numerator is the total number of sentences in the considered news and the denominator is the number of sentences where the considered word appears. *Cosine* means cosine similarity measure between two sentence vectors.

4.4 Summarizer

Our summarizer contains the preprocessing stage and the estimating stage. The preprocessing stage extracts features and normalizes all features by equation (1).

$$N_j = \frac{w_j - mean(w_j)}{dev(w_j)} \tag{1}$$

Here, w_j is the original value of feature j which is used to describe sentence i; $mean(w_j)$ is the mean value of feature j in our training set or test set; $dev(w_j)$ is the standard deviation value of feature j in our training set or test set.

The estimating stage predicts whether each sentence of the broadcast news is in a summary or not. We use Radial Basis Function(RBF) kernel for constructing SVM classifier as our estimator referring to LIBSVM (Chang and Lin, 2001), which is a library for support vector machines.

5 Experiments and Evaluation

We use the several-turn news corpus, described in Section 3, in our experiments. We use 70% of the corpus consisting of 3294 sentences as training set and the remaining 1454 sentences as held-out test set, upon which our summarizer is tested.

Table 1: Feature set Evaluation by F-measure

Feature Set	SR10%	SR15%	SR20%	Ave
Ac+St+Le	.5961	.546	.5544	.5655
Ac+St	**.5888**	**.5489**	**.5562**	**.5646**
St	.5951	.5616	.537	.5645
Le	.5175	.5219	.5329	.5241
Ac	.3068	.4092	.4582	.3914
Baseline	.21	.32	.43	.32

Ac: Acoustic; St: Structural; Le: Lexical

We measure our summarizer's performance by precision, recall, and F-measure (Jing et al., 1998). We explain these metrics as follows:

$$\text{precision} = \frac{S_{man} \bigcap S_{sum}}{S_{sum}} \tag{2}$$

$$\text{recall} = \frac{S_{man} \bigcap S_{sum}}{S_{man}} \tag{3}$$

$$\text{F-Measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{4}$$

In equation (2), (3) and (4), S_{man} is the sentence set of manual summaries or reference summaries; S_{sum} is the sentence set of predicted summaries provided by our summarizer.

We have three versions of reference summaries based on summarization ratio(SR): 10%, 15% and 20% respectively. So we build three baselines referring to different versions of reference summaries. When using SR 10% summaries, we build the baselines by choosing the first 10% of sentences from each story. Our baseline results in F-measure score are given in Table 1.

We perform three sets of experiments with different summarization ratios.

By using acoustic and structural features alone, the summarizer produces the same performance as by using all features. We can find the evidence from Table 1 and Figure 1. On average, the combination of acoustic and structural features yields good performance: F-measure of 0.5646, 24.46% higher than the baseline, only 0.09% lower than the average F-measure produced by using all features. This finding makes it possible to summarize speech without

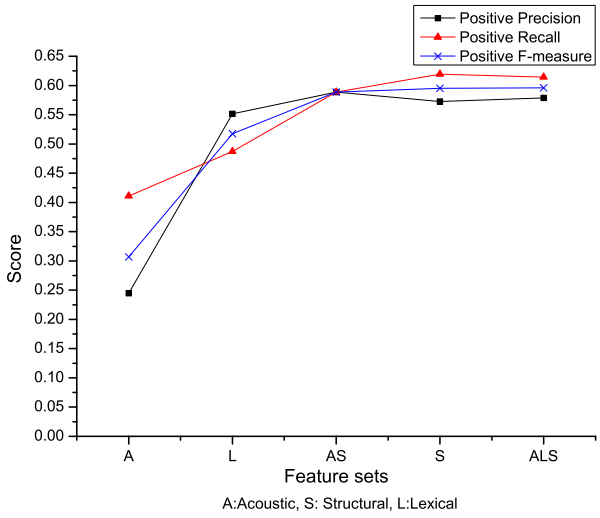


Figure 1: Performance comparison on SR10%

placing a stringent demand on the speech recognition accuracy.

In the same Mandarin broadcast program, the distribution and flow of summary sentences are relatively consistent. Therefore, compared with speech summarization on English sources, we can achieve the different finding that structural features play a key role in speech summarization for Mandarin broadcast news. Table 1 shows the evidence. On average, structural features are superior to lexical features: F-measure of 0.5645, 24.45% higher than the baseline and 4,04% higher than the average F-measure produced by using lexical features.

Another conclusion we can draw from Table 1 is that acoustic features are important for speech summarization on Mandarin broadcast news. On average, even by using acoustic features alone our summarizer yields competitive result: F-measure of 0.3914, 7.14% higher than the baseline. The similar conclusion also holds for speech summarization on English sources (Maskey and Hirschberg, 2006).

6 Conclusion

In this paper, we have presented the results of an empirical study on speech summarization for Mandarin broadcast news. From these results, we found that by using acoustic and structural features alone, the summarizer produces good performance: average F-measure of 0.5646, the same as by using all features. We also found that structural features make more important contribution than lexical features to speech summarization because of the relatively consistent distribution and flow of summary sentences in the same Mandarin broadcast program. Moreover, we have shown that our summarizer performed surprisingly well by using only acoustic features: average F-measure of 0.3914, 7.14% higher than the baseline. These findings also suggest that high quality speech summarization can be achieved without stringent requirement on speech recognition accuracy.

References

P. Boersma and D. Weenink. 1996. Praat, a system for doing phonetics by computer, version 3.4. *Institute of Phonetic Sciences of the University of Amsterdam, Report*, 132:182.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*.

B. Chen, Y.M. Yeh, Y.M. Huang, and Y.T. Chen. 2006. Chinese Spoken Document Summarization Using Probabilistic Latent Topical Information. *Proc. ICASSP*.

J. Hirschberg. 2002. Communication and prosody: Functional aspects of prosody. *Speech Communication*, 36(1):31–43.

C. Hori and S. Furui. 2003. A new approach to automatic speech summarization. *Multimedia, IEEE Transactions on*, 5(3):368–378.

C.L. Huang, C.H. Hsieh, and C.H. Wu. 2005. Spoken Document Summarization Using Acoustic, Prosodic and Semantic Information. *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 434–437.

A. Inoue, T. Mikami, and Y. Yamashita. 2004. Improvement of Speech Summarization Using Prosodic Information. *Proc. of Speech Prosody*.

H. Jing, R. Barzilay, K. McKeown, and M. Elhadad. 1998. Summarization evaluation methods: Experiments and analysis. *AAAI Symposium on Intelligent Summarization*.

K. Koumpis and S. Renals. 2005. Automatic summarization of voicemail messages using lexical and prosodic features. *ACM Transactions on Speech and Language Processing (TSLP)*, 2(1):1–24.

S. Maskey and J. Hirschberg. 2003. Automatic summarization of broadcast news using structural features. *Proceedings of Eurospeech 2003*.

S. Maskey and J. Hirschberg. 2005. Comparing lexical, acoustic/prosodic, structural and discourse features for speech summarization. *Interspeech 2005 (Eurospeech)*.

S. Maskey and J. Hirschberg. 2006. Summarizing Speech Without Text Using Hidden Markov Models. *Proc. NAACL*.

A Semi-Automatic Evaluation Scheme: Automated Nuggetization for Manual Annotation

Liang Zhou, Namhee Kwon, and Eduard Hovy
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292
{liangz, nkwon, hovy}@isi.edu

Abstract

In this paper we describe automatic information nuggetization and its application to text comparison. More specifically, we take a close look at how machine-generated nuggets can be used to create evaluation material. A semi-automatic annotation scheme is designed to produce gold-standard data with exceptionally high inter-human agreement.

1 Introduction

In many natural language processing (NLP) tasks, we are faced with the problem of determining the appropriate granularity level for information units. Most commonly, we use sentences to model individual pieces of information. However, more NLP applications require us to define text units smaller than sentences, essentially decomposing sentences into a collection of phrases. Each phrase carries an independent piece of information that can be used as a standalone unit. These finer-grained information units are usually referred to as *nuggets*.

When performing within-sentence comparison for redundancy and/or relevancy judgments, without a precise and consistent breakdown of nuggets we can only rely on rudimentary *n*-gram segmentations of sentences to form nuggets and perform subsequent *n*-gram-wise text comparison. This is not satisfactory for a variety of reasons. For example, one *n*-gram window may contain several separate pieces of information, while another of the same length may not contain even one complete piece of information.

Previous work shows that humans can create nuggets in a relatively straightforward fashion. In the PYRAMID scheme for manual evaluation of summaries (Nenkova and Passonneau, 2004), machine-generated summaries were compared with human-written ones at the nugget level. However, automatic creation of the nuggets is not trivial. Hamly et al. (2005) explore the enumeration and combination of all words in a sentence to create the set of all possible nuggets. Their automation process still requires nuggets to be manually created *a priori* for reference summaries before any summary comparison takes place. This human involvement allows a much smaller subset of phrase segments, resulting from word enumeration, to be matched in summary comparisons. Without the human-created nuggets, text comparison falls back to its dependency on n-grams. Similarly, in question-answering (QA) evaluations, gold-standard answers use manually created nuggets and compare them against system-produced answers broken down into n-gram pieces, as shown in POURPRE (Lin and Demner-Fushman, 2005) and NUGGETEER (Marton and Radul, 2006).

A serious problem in manual nugget creation is the inconsistency in human decisions (Lin and Hovy, 2003). The same nugget will not be marked consistently with the same words when sentences containing multiple instances of it are presented to human annotators. And if the annotation is performed over an extended period of time, the consistency is even lower. In recent exercises of the PYRAMID evaluation, inconsistent nuggets are flagged by a tracking program and returned back to the annotators, and resolved manually.

Given these issues, we address two questions in this paper: First, how do we define nuggets so that they are consistent in definition? Secondly, how do

Proceedings of NAACL HLT 2007, Companion Volume, pages 217–220,
Rochester, NY, April 2007.

we utilize automatically extracted nuggets for various evaluation purposes?

2 Nugget Definition

Based on our manual analysis and computational modeling of nuggets, we define them as follows:
Definition:

- A nugget is predicated on either an *event* or an *entity*.
- Each nugget consists of two parts: the anchor and the content.

The anchor is either:

- the head noun of the entity, or
- the head verb of the event, plus the head noun of its associated entity (if more than one entity is attached to the verb, then its subject).

The content is a coherent single piece of information associated with the anchor. Each anchor may have several separate contents.

When a nugget contains nested sentences, this definition is applied recursively. Figure 1 shows an example. Anchors are marked with square brackets. If the anchor is a verb, then its entity attachment is marked with curly brackets. If the sentence in question is a compound and/or complex sentence, then this definition is applied recursively to allow decomposition. For example, in Figure 1, without recursive decomposition, only two nuggets are formed: 1) "[girl] working at the bookstore in Hollywood", and 2) "{girl} [talked] to the diplomat living in Britain". In this example, recursive decomposition produces nuggets with labels 1-a, 1-b, 2-a, and 2-b.

2.1 Nugget Extraction

We use syntactic parse trees produced by the Collins parser (Collins, 1999) to obtain the structural representation of sentences. Nuggets are extracted by identifying subtrees that are descriptions for entities and events. For entity nuggets, we examine subtrees headed by "NP"; for event nuggets, subtrees headed by "VP" are examined and their corresponding subjects (siblings headed by "NP") are treated as entity attachments for the verb phrases.

Sentence:
The girl working at the bookstore in Hollywood talked to the diplomat living in Britain.

Nuggets are:

1) [girl] working at the bookstore in Hollywood
    a. [girl] working at the bookstore
    b. [bookstore] in Hollywood
2) {girl} [talked] to the diplomat living in Britain
    a. {girl} [talked] to the diplomat
    b. [diplomat] living in Britian

Anchors:

1) [girl]
    a. [girl]
    b. [bookstore]
2) {girl} [talked]: talked is the anchor verb and girl is its entity attachment.
    a. {girl} [talked]
    b. [diplomat]

Figure 1. Nugget definition examples.

3 Utilizing Nuggets in Evaluations

In recent QA and summarization evaluation exercises, manually created nuggets play a determinate role in judging system qualities. Although the two task evaluations are similar, the text comparison task in summarization evaluation is more complex because systems are required to produce long responses and thus it is hard to yield high agreement if manual annotations are performed. The following experiments are conducted in the realm of summarization evaluation.

3.1 Manually Created Nuggets

During the recent two Document Understanding Confereces (DUC-05 and DUC-06) (NIST, 2002–2007), the PYRAMID framework (Nenkova and Passonneau, 2004) was used for manual summary evaluations. In this framework, human annotators select and highlight portions of reference summaries to form a pyramid of summary content units (SCUs) for each docset. A pyramid is constructed from SCUs and their corresponding popularity scores—the number of reference summaries they appeared in individually. SCUs carrying the same information do not necessarily have the same surface-level words. Annotators need to make the decisions based on semantic equivalence among

various SCUs. To evaluate a peer summary from a particular docset, annotators highlight portions of text in the peer summary that convey the same information as those SCUs in previously constructed pyramids.

3.2 Automatically Created Nuggets

We envisage the nuggetization process being automated and nugget comparison and aggregation being performed by humans. It is crucial to involve humans in the evaluation process because recognizing semantically equivalent units is not a trivial task computationally. In addition, since nuggets are system-produced and can be imperfect, annotators are allowed to reject and re-create them. We perform record-keeping in the background on which nugget or nugget groups are edited so that further improvements can be made for nuggetization.

The evaluation scheme is designed as follows:

For *reference summaries* (per docset):

- Nuggets are created for all sentences;
- Annotators will group equivalent nuggets.
- Popularity scores are automatically assigned to nugget groups.

For *peer summaries*:

- Nuggets are created for all sentences;
- Annotators will match/align peer's nuggets with reference nugget groups.
- Recall scores are to be computed.

3.3 Consistency in Human Involvement

The process of creating nuggets has been automated and we can assume a certain level of consistency based on the usage of the syntactic parser. However, a more important issue emerges. When given the same set of nuggets, would human annotators agree on nugget group selections and their corresponding contributing nuggets? What levels of agreement and disagreement should be expected? Two annotators, one familiar with the notion of nuggetization (C1) and one not (C2), participated in the following experiments.

Figure 2 shows the annotation procedure for reference summaries. After two rounds of individual annotations and consolidations and one final round of conflict resolution, a set of gold-standard

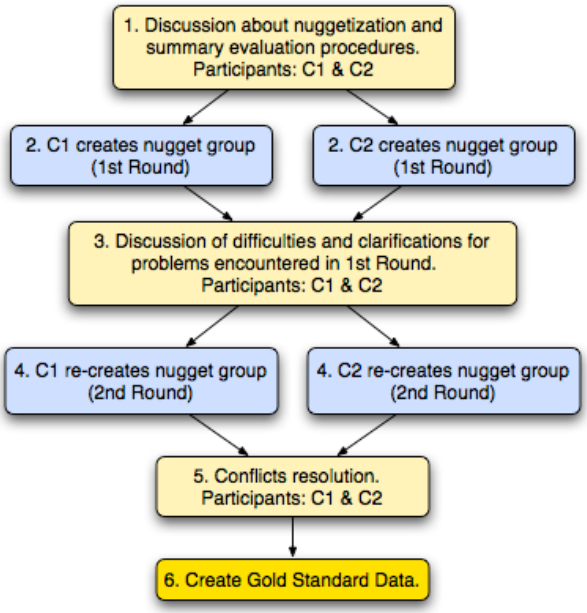


Figure 2. Reference annotation and gold-standard data creation.

nugget groups is created for each docset and will be subsequently used in peer summary annotations. The first round of annotation is needed since one of the annotators, C2, is not familiar with nuggetization. After the initial introduction of the task, concerns and questions arisen can be addressed. Then the annotators proceed to the second round of annotation. Naturally, some differences and conflicts remain. Annotators must resolve these problems during the final round of conflict resolution and create the agreed-upon gold-standard data. Previous manual nugget annotation has used one annotator as the primary nugget creator and another annotator as an inspector (Nenkova and Passonneau, 2004). In our annotation experiment, we encourage both annotators to play equally active roles. Conflicts between annotators resulting from ideology, comprehension, and interpretation differences helped us to understand that complete agreement between annotators is not realistic and not achievable, unless one annotator is dominant over the other. We should expect a 5-10% annotation variation.

In Figure 3, we show annotation comparisons from first to second round. The *x*-axis shows the nugget groups that C1 and C2 have agreed on. The *y*-axis shows the popularity score a particular nugget group received. Selecting from three reference summaries, a score of three for a nugget group indicates it was created from nuggets in all three

summaries. The first round initially appears successful because the two annotators had 100% agreement on nugget groups and their corresponding scores. However, C2, the novice nuggetizer, was much more conservative than C1, because only 10 nugget groups were created. The geometric mean of agreement on all nugget group assignment is merely 0.4786. During the second round, differences in group-score allocations emerge, 0.9192, because C2 is creating more nugget groups. The geometric mean of agreement on all nugget group assignment has been improved to 0.7465.

After the final round of conflict resolution, gold-standard data was created. Since all conflicts must be resolved, annotators have to either convince or be convinced by the other. How much change is there between an annotator's second-round annotation and the gold-standard? Geometric mean of agreement on all nugget group assignment for C1 is 0.7543 and for C2 is 0.8099. Agreement on nugget group score allocation for C1 is 0.9681 and for C2 is 0.9333. From these figures, we see that while C2 contributed more to the gold-standard's nugget group creations, C1 had more accuracy in finding the correct number of nugget occurrences in reference summaries. This confirms that both annotators played an active role. Using the gold-standard nugget groups, the annotators performed 4 peer summary annotations. The agreement among peer summary annotations is quite high, at approximately 0.95. Among the four, annotations on one peer summary from the two annotators are completely identical.

4 Conclusion

In this paper we have given a concrete definition for information nuggets and provided a systematic implementation of them. Our main goal is to use these machine-generated nuggets in a semi-automatic evaluation environment for various NLP applications. We took a close look at how this can be accomplished for summary evaluation, using nuggets created from reference summaries to grade peer summaries. Inter-annotator agreements are measured to insure the quality of the gold-standard data created. And the agreements are very high by following a meticulous procedure. We are currently preparing to deploy our design into full-scale evaluation exercises.

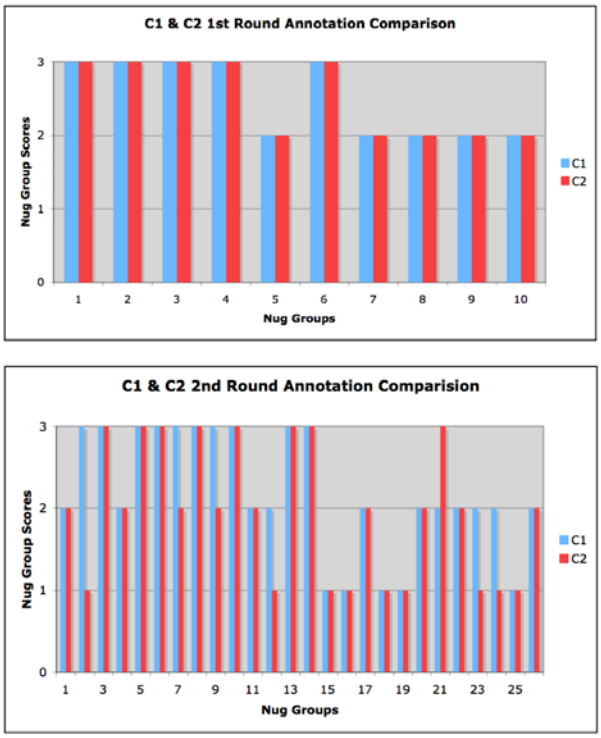


Figure 3. Annotation comparisons from 1st to 2nd round.

References

Collins, M. 1999. Head-driven statistical models for natural language processing. *PhD Dissertation*, University of Pennsylvania.

Hamly, A., A. Nenkova, R. Passonneau, and O. Rambow. 2005. Automation of summary evaluation by the pyramid method. In *Proceedings of* RANLP.

Lin, C.Y. and E. Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of* NAACL-HLT.

Lin, J. and D. Demner-Fushman. 2005. Automatically evaluating answers to definition questions. In *Proceedings of* HLT-EMNLP.

Marton, G. and A. Radul. 2006. Nuggeteer: automatic nugget-based evaluation using description and judgments. In *Proceedings* NAACL-HLT.

Nenkova, A. and R. Passonneau. 2004. Evaluating content selection in summarization: the pyramid method. In *Proceedings* NAACL-HLT.

NIST. 2001–2007. Document Understanding Conference. www-nlpir.nist.gov/projects/duc/index.html.

Author Index